Tell us about your PDF experience.

Windows deployment documentation

Learn about deploying and updating Windows devices in your organization.

Plan

CONCEPT

Plan for Windows 11

Create a deployment plan

Plan for volume activation

Windows compatibility cookbook

Cloud-native Windows

Prepare

GET STARTED

Prepare for Windows 11

Prepare to deploy Windows updates

Prepare for Windows Update client policies

Evaluate and update infrastructure

Deploy

DEPLOY

Compare Windows Autopilot solutions

Deploy updates with Intune

Deploy Windows updates with Configuration Manager

Optimize and cache content

Use Windows Autopilot

OVERVIEW

Windows Autopilot device preparation overview

Windows Autopilot overview

TUTORIAL

Windows Autopilot scenarios

Windows Autopilot device preparation scenarios

Use Windows Autopatch

OVERVIEW

What is Windows Autopatch?

Prerequisites

Deployment guide

See more >

Use Windows Update client policies

HOW-TO GUIDE

What are Windows Update client policies?

Configure Windows Update client policies

Windows Update for Business reports overview

Use tools for upgrade and imaging

REFERENCE

Customize Windows PE boot images

Convert a disk from MBR to GPT

Configure a PXE server to load Windows PE

Resolve Windows upgrade errors

More resources

DOWNLOAD

Download and install the Windows ADK

Deployment tools

WHAT'S NEW

Windows IT pro blog

Windows office hours

Windows deployment scenarios

Article • 02/27/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

To successfully deploy the Windows operating system in an organization, it's important to understand the different ways that it can be deployed. Key tasks include choosing among these scenarios and understanding the capabilities and limitations of each.

Deployment categories

The following tables summarize various Windows deployment scenarios. The scenarios are each assigned to one of three categories.

- Modern deployment methods are recommended unless a specific need requires use of a different procedure. These methods are supported with existing tools such as Microsoft Configuration Manager.

  > **Note**
  >
  > Once Windows is deployed in an organization, it's important to stay up to date by creating a deployment plan for Windows feature updates.

- Dynamic deployment methods enable configuration of applications and settings for specific use cases.
- Traditional deployment methods use existing tools to deploy operating system images.

Modern

Expand table

Scenario	Description	More information
Windows Autopilot	Customize the out-of-box-experience (OOBE) for an organization, and deploy a new system with apps and settings already configured	Overview of Windows Autopilot
In-place upgrade	Use Windows Setup to update the Windows version and migrate apps and settings. Rollback data is saved in Windows.old.	Perform an in-place upgrade to Windows using Configuration Manager

Dynamic

Expand table

Scenario	Description	More information
Subscription Activation	Switch from Windows Pro to Enterprise when a subscribed user signs in.	Windows Subscription Activation
Microsoft Entra ID / MDM	The device is automatically joined to Microsoft Entra ID and configured by MDM.	Microsoft Entra integration with MDM
Provisioning packages	Using the Windows Imaging and Configuration Designer tool, create provisioning packages that can be applied to devices.	Configure devices without MDM

Traditional

Expand table

Scenario	Description	More information
Bare metal	Deploy a new device, or wipe an existing device and deploy with a fresh image.	Deploy Windows using PXE and Configuration Manager
Refresh	Also called wipe and load. Redeploy a device by saving the user state, wiping the disk, then restoring the user state.	Refresh a Windows client with a currently supported version of Windows using Configuration Manager
Replace	Replace an existing device with a new one by saving the user state on the old device and then restoring it to the new device.	Replace a Windows client with a currently supported version of Windows using Configuration Manager

Important

The Windows Autopilot and Subscription Activation scenarios require that the beginning OS is a currently supported version of Windows.

Except for clean install scenarios such as traditional bare metal and Windows Autopilot, all the methods described can optionally migrate apps and settings to the new OS.

Modern deployment methods

Modern deployment methods embrace both traditional on-premises and cloud services to deliver a streamlined and cost effective deployment experience.

Windows Autopilot

Windows Autopilot is a new suite of capabilities designed to simplify and modernize the deployment and management of new Windows PCs. Windows Autopilot enables IT professionals to customize the Out of Box Experience (OOBE) for Windows PCs and provide end users with a fully configured new Windows device. There are no images to deploy, no drivers to inject, and no infrastructure to manage. Users can go through the deployment process independently, without the need consult their IT administrator.

For more information about Windows Autopilot, see Overview of Windows Autopilot and Modernizing Windows deployment with Windows Autopilot.

In-place upgrade

For existing computers running out of support versions of Windows, the recommended path for organizations deploying Windows is to perform an in-place upgrade. An in-place upgrade uses the Windows installation program (`Setup.exe`) to:

- Automatically preserves all data, settings, applications, and drivers from the existing operating system version
- Requires the least IT effort, because there's no need for any complex deployment infrastructure

Although consumer PCs are upgraded using Windows Update, organizations want more control over the process. Control is accomplished by using tools like Microsoft Configuration Manager to completely automate the upgrade process through simple task sequences.

The in-place upgrade process is designed to be reliable. An in-place upgrade has the ability to automatically roll back to the previous operating system if any issues are encountered during the deployment process, without any IT staff involvement. Rolling back manually can also be done by using the automatically created recovery information (stored in the Windows.old folder), in case any issues are encountered after the upgrade is finished. The upgrade process is also typically faster than traditional deployments, because applications don't need to be reinstalled as part of the process.

Existing applications are preserved through the process. The upgrade process uses the standard Windows installation media image (Install.wim). Custom images not only aren't needed, but they also can't be used. Custom images can't be used because the upgrade process is unable to deal with conflicts between apps in the old and new operating system. For example, Contoso Timecard 1.0 in Windows 10 and Contoso Timecard 3.0 in the Windows 11 image.

Scenarios that support in-place upgrade with some other procedures include changing from BIOS to UEFI boot mode and upgrade of devices that use non-Microsoft disk encryption software.

- **Legacy BIOS to UEFI booting**: To perform an in-place upgrade on a UEFI-capable system that currently boots using legacy BIOS, first perform the in-place upgrade to Windows 10, maintaining the legacy BIOS boot mode. Windows 10 doesn't require UEFI, so it works fine to upgrade a system using legacy BIOS emulation. After the upgrade, the system disk can be converted to a format that supports UEFI boot using the MBR2GPT tool. UEFI specification requires GPT disk layout. After the disk is converted, the firmware of the device must also be configured to boot in UEFI mode. Enabling UEFI also UEFI features such as Secure Boot to be enabled.

> **Important**
>
> Performing an in-place upgrade on a UEFI-capable system that currently boots using legacy BIOS is only possible with Windows 10. Windows versions newer than Windows 10 only support UEFI-capable systems and don't support legacy BIOS or MBR.

- **Non-Microsoft disk encryption software**: While devices encrypted with BitLocker can easily be upgraded, more work is necessary for non-Microsoft disk encryption tools. Some ISVs might provide instructions on how to integrate their software into the in-place upgrade process. Check with the ISV to see if they have instructions. The following articles provide details on how to provision encryption drivers for use during Windows Setup via the ReflectDrivers setting:
  - Windows Setup Automation Overview
  - Windows Setup Command-Line Options

There are some situations where an in-place upgrade can't be used. In these situations, use traditional deployment methods instead. Examples of these situations include:

- Changing from an x86 version of Windows 10 to an x64 version of Windows. Versions of Windows newer than Windows 10 are only x64 and don't have an x86 version. The upgrade process can't change from a 32-bit operating system to a 64-

bit operating system, because of possible complications with installed applications and drivers.

- Boot from VHD installations. The upgrade process is unable to upgrade these installations. Instead, new installations would need to be performed.
- Updating existing images. It can be tempting to try to upgrade existing Windows images to a newer version of Windows by installing the old image, upgrading it, and then recapturing the new Windows image. However, this scenario isn't supported. Preparing an upgraded OS via `Sysprep.exe` before capturing an image isn't supported and doesn't work. When `Sysprep.exe` detects the upgraded OS, it fails.
- Dual-boot and multi-boot systems. The upgrade process is designed for devices running a single OS. If using dual-boot or multi-boot systems with multiple operating systems, then extra care should be taken. Dual-boot and multi-boot systems doesn't include using virtual machines for the second and subsequent operating systems.

Dynamic provisioning

For new PCs, organizations historically replaced the version of Windows included on the device with their own custom Windows image. A custom image was used because a custom image was often faster and easier than using the preinstalled version. However, reimaging with a custom image is an added expense due to the time and effort required. With the new dynamic provisioning capabilities and tools provided with Windows, it's now possible to avoid using custom images.

The goal of dynamic provisioning is to take a new PC out of the box, turn it on, and transform it into a productive organization device, with minimal time and effort. The types of transformations that are available include:

Windows Subscription Activation

Windows Subscription Activation is a dynamic deployment method that enables changing the edition of Windows from Pro to Enterprise. Windows Subscription Activation requires no keys and no reboots. For more information about Subscription Activation, see Windows Subscription Activation.

Microsoft Entra join with automatic mobile device management (MDM) enrollment

In this scenario, the organization member just needs to provide their work or school user ID and password. The device can then be automatically joined to Microsoft Entra ID and enrolled in a mobile device management (MDM) solution with no other user interaction. Once done, the MDM solution can finish configuring the device as needed. For more information, see Microsoft Entra integration with MDM.

Provisioning package configuration

With the Windows Imaging and Configuration Designer (ICD), IT administrators can create a self-contained package that contains all of the configuration, settings, and apps that need to be applied to a device. These packages can then be deployed to new PCs through various means, typically by IT professionals. For more information, see Provisioning packages for Windows.

These scenarios can be used to enable "Bring Your Own Device" (BYOD) or "Choose Your Own Device" (CYOD) programs. With these programs, an organization's users can pick their own PC. They aren't restricted to a small list of approved or certified models. These programs are difficult to implement using traditional deployment scenarios.

While Windows includes various provisioning settings and deployment mechanisms, provisioning settings and deployment mechanisms continue to be enhanced and extended based on feedback from organizations. As with all Windows features, organizations can submit suggestions for more features through the Windows Feedback app or through their Microsoft Support contacts.

Traditional deployment

In the past, organizations typically deployed Windows using an image-based process built on top of tools provided in:

- Windows Assessment and Deployment Kit.
- Microsoft Configuration Manager.
- Windows Deployment Services (WDS).
- Microsoft Deployment Toolkit.

Scenarios such as in-place upgrade and dynamic provisioning might reduce the need for traditional deployment capabilities in some organizations. However, traditional methods might still need to be used under certain circumstances.

The traditional deployment scenario can be divided into different sub-scenarios. These sub-scenarios are explained in detail in the following sections, but the following list provides a brief summary:

- **New computer**: A bare-metal deployment of a new device.
- **Computer refresh**: A reinstall of the same device (with user-state migration and an optional full Windows Imaging (WIM) image backup).
- **Computer replace**: A replacement of the old device with a new device (with user-state migration and an optional full WIM image backup).

New computer

Also called a "bare metal" deployment. This scenario occurs when there's a device with no OS installed on it that needs to be deployed. This scenario can also be an existing device that needs to be wiped and redeployed without needing to preserve any existing data. The setup starts from a boot media, using CD, USB, ISO, or Pre-Boot Execution Environment (PXE). A full offline media that includes all the files needed for a client deployment can also be generated, allowing deployment without having to connect to a central deployment share. The target can be a physical computer, a virtual machine, or a Virtual Hard Disk (VHD) running on a physical computer (boot from VHD).

The deployment process for the new device scenario is as follows:

1. Start the setup from boot media (CD, USB, ISO, or PXE).

2. Wipe the hard disk clean and create new volume(s).

3. Install the operating system image.

4. Install other applications (as part of the task sequence).

After following these steps, the computer is ready for use.

Computer refresh

A refresh is sometimes called wipe-and-load. The process is normally initiated in the running operating system. User data and settings are backed up and restored later as part of the deployment process. The target can be the same as for the new computer scenario.

The deployment process for the wipe-and-load scenario is as follows:

1. Start the setup on a running operating system.

2. Save the user state locally.

3. Wipe the hard disk clean (except for the folder containing the backup).

4. Install the operating system image.
5. Install other applications.
6. Restore the user state.

After following these steps, the device is ready for use.

Computer replace

A computer replace is similar to the refresh scenario. However, since we're replacing the device, we divide this scenario into two main tasks: backup of the old client and bare-metal deployment of the new client. As with the refresh scenario, user data and settings are backed up and restored.

The deployment process for the replace scenario is as follows:

1. Save the user state (data and settings) on the server through a backup job on the running operating system.
2. Deploy the new computer as a bare-metal deployment.

> **Note**
>
> In some situations, the replace scenario can be used even if the target is the same device. For example, replace can be used if disk layout needs to be changed from master boot record (MBR) to GUID partition table (GPT). This conversion allows taking advantage of Unified Extensible Firmware Interface (UEFI) functionality.

Related articles

- Upgrade to Windows with Microsoft Configuration Manager.
- Deploy Windows using PXE and Configuration Manager.
- Windows setup technical reference.
- Windows Imaging and Configuration Designer.
- UEFI firmware.

Feedback

Was this page helpful? Yes No

Provide product feedback

Quick guide to Windows as a service

Article • 07/17/2024 • Applies to: ✅ Windows 11, ✅ Windows 10

Here's a quick guide to the most important concepts in Windows as a service.

Definitions

Some new terms have been introduced as part of Windows as a service, so you should know what these terms mean.

- **Feature updates** are released annually. As the name suggests, these updates add new features, delivered in bite-sized chunks compared to the previous practice of Windows releases every 3-5 years.
- **Quality updates** deliver both security and nonsecurity fixes. They're typically released on the second Tuesday of each month, though they can be released at any time. Quality updates include security updates, critical updates, servicing stack updates, and driver updates. Quality updates are cumulative, so installing the latest quality update is sufficient to get all the available fixes for a specific Windows 10 feature update. The "servicing stack" is the code that installs other updates, so they're important to keep current. For more information, see Servicing stack updates.
- **Insider Preview** builds are made available during the development of the features that will be shipped in the next feature update, enabling organizations to validate new features and confirm compatibility with existing apps and infrastructure, providing feedback to Microsoft on any issues encountered.
- **Servicing channels** allow organizations to choose when to deploy new features.
  - The **General Availability Channel** receives feature updates annually.
  - The **Long-Term Servicing Channel**, which is meant only for specialized devices (which typically don't run Office) such as those that control medical equipment or ATMs, receives new feature releases every two to three years.
- **Deployment rings** are groups of devices used to initially pilot, and then to broadly deploy, each feature update in an organization.

For more information, see Overview of Windows as a service.

For some interesting in-depth information about how cumulative updates work, see Windows Updates using forward and reverse differentials.

Key concepts

With each release in the General Availability Channel, we recommend beginning deployment right away to devices selected for early adoption (targeted validation) and ramp up to full deployment at your discretion.

Windows Enterprise LTSC versions are separate **Long-Term Servicing Channel** versions. Each release is supported for a total of 10 years (five years standard support, five years extended support). New releases are expected about every three years.

For more information, see Assign devices to servicing channels for Windows client updates.

Staying up to date

To stay up to date, deploy feature updates at an appropriate time after their release. You can use various management and update tools such as Windows Update, Windows Update client policies, Windows Server Update Services, Microsoft Configuration Manager, and non-Microsoft products to help with this process.

Extensive advanced testing isn't required. Instead, only business-critical apps need to be tested, with the remaining apps validated through a series of pilot deployment rings. Once these pilot deployments have validated most apps, broad deployment can begin.

This process repeats with each new feature update. These are small deployment projects, compared to the large projects that were necessary with the old three-to-five-year Windows release cycles.

Other technologies such as BranchCache and Delivery Optimization, both peer-to-peer distribution tools, can help with the distribution of the feature update installation files.

Feedback

Was this page helpful? Yes No

Provide product feedback

Overview of Windows as a service

Applies to: ✅ Windows 11, ✅ Windows 10

> **Looking for consumer information?** See Windows Update: FAQ.

Windows as a service is a way to simplify the lives of IT pros and maintain a consistent Windows experience for its customers. These improvements focus on maximizing customer involvement in Windows development, simplifying the deployment and servicing of Windows client computers, and leveling out the resources needed to deploy and maintain Windows over time.

Building

Prior to Windows 10, Microsoft released new versions of Windows every few years. This traditional deployment schedule imposed a training burden on users because the feature revisions were often significant. That schedule also meant waiting long periods without new features. That scenario doesn't always work in today's rapidly changing world, a world in which new security, management, and deployment capabilities are necessary to address challenges.

In the past, when Microsoft developed new versions of Windows, it typically released technical previews near the end of the process, when Windows was nearly ready to ship. Now, new features are delivered to the Windows Insider Program as soon as possible, during the development cycle, through a process called *flighting*. Organizations can see exactly what Microsoft is developing and start their testing as soon as possible.

Microsoft also depends on receiving feedback from organizations throughout the development process so that it can make adjustments as quickly as possible rather than waiting until after release. For more information about the Windows Insider Program and how to sign up, see the section Windows Insider.

Microsoft also runs extensive internal testing, with engineering teams installing new builds daily, and larger groups of employees installing builds frequently, all before those builds are ever released to the Windows Insider Program.

Deploying

Deploying a modern version of Windows is simpler than with previous versions of Windows. When migrating from earlier versions of Windows, you can use an easy in-place upgrade process to automatically preserve all apps, settings, and data. Afterwards, deployment of feature updates is equally simple.

Application compatibility

Application compatibility testing has historically been a burden when approaching a Windows deployment or upgrade. Application compatibility from the perspective of desktop applications, websites, and apps built on the Universal Windows Platform (UWP) has improved tremendously over older versions of Windows. For the most important business-critical applications, organizations should still perform testing regularly to validate compatibility with new builds.

Servicing

Traditional Windows servicing has included several release types: major revisions (for example, the Windows 8.1, Windows 8, and Windows 7 operating systems), service packs, and monthly updates. With Windows 10 and Windows 11, there are two release types:

- **Quality updates**: Updates that provide security and reliability fixes and can also add new functionality or features periodically.
- **Feature updates**: Updates that are released once a year on which the servicing timeline for the operating system is based. These updates can contain new features and functionality and previously released quality updates.

Servicing channels are the first way to separate users into deployment groups for feature and quality updates. For more information about developing a deployment strategy that uses servicing channels, see Plan servicing strategy for Windows client updates.

For information about each servicing tool, see Servicing tools.

There are three servicing channels, each of which provides different levels of flexibility over when these updates are delivered to client computers. For more information, see Servicing channels.

There are currently three release channels for Windows clients:

- The **General Availability Channel** receives feature updates as soon as they're available.
- The **Long-Term Servicing Channel** (LTSC) is designed to be used only for specialized devices such as those that control medical equipment or ATM machines. LTSC receives new feature releases every two to three years.
- The **Windows Insider Program** provides organizations with the opportunity to test and provide feedback on features that will be shipped in upcoming updates.

With each General Availability release, you would typically begin deploying right away to devices selected for early adoption (targeted validation) and then ramp up to full deployment

at your discretion. This enables you to gain access to new features, experiences, and integrated security as soon as possible.

> **Important**
>
> Devices on the General Availability Channel must have their diagnostic data set to **1 (Basic)** or higher in order to ensure that the service is performing at the expected quality. For instructions to set the diagnostic data level, see Configure Windows diagnostic data in your organization.

Feature updates

You can deploy feature updates using existing management tools. These changes come in bite-sized chunks rather than all at once like in previous versions of Windows, decreasing user readiness time.

Quality updates

Monthly updates in previous Windows versions were often overwhelming because of the sheer number of updates available each month. Many organizations selectively chose which updates they wanted to install and which they didn't, and this created countless scenarios in which organizations deployed essential security updates but picked only a subset of nonsecurity fixes.

Rather than receiving several updates each month and trying to figure out which the organization needs, which ultimately causes platform fragmentation, administrators see one cumulative monthly update that supersedes the previous month's update, containing both security and nonsecurity payloads. This approach makes updating simpler and ensures that devices are more closely aligned with the testing done at Microsoft, reducing unexpected issues resulting from updates.

Servicing channels

There are three servicing channels. The Windows Insider Program provides organizations with the opportunity to test and provide feedback on features that will be shipped in the next feature update. The General Availability Channel provides new functionality with feature update releases. Organizations can choose when to deploy updates from the General Availability Channel. The Long-Term Servicing Channel, which is designed to be used only for specialized devices (which typically don't run Office) such as those that control medical equipment or ATM machines, receives new feature releases every two to three years. For more information about the versions in each servicing channel, see Windows release information.

Servicing channels aren't the only way to separate groups of devices when consuming updates. Each channel can contain subsets of devices, which staggers servicing even further. For information about the servicing strategy and ongoing deployment process for Windows 10, including the role of servicing channels, see Plan servicing strategy for Windows client updates.

General Availability Channel

In the General Availability Channel, feature updates are available annually. This servicing model is ideal for pilot deployments and testing of feature updates and for users such as developers who need to work with the latest features. Once the latest release has gone through pilot deployment and testing, you'll be able to choose the timing at which it goes into broad deployment.

When Microsoft officially releases a feature update, we make it available to any device not configured to defer feature updates so that those devices can immediately install it. Organizations that use Windows Server Update Services (WSUS), Microsoft Configuration Manager, or Windows Update client policies, however, can defer feature updates to selective devices by withholding their approval and deployment. In this scenario, the content available for the General Availability Channel is available but not necessarily immediately mandatory, depending on the policy of the management system. Organizations can electively delay feature updates into as many phases as they wish by using servicing tools. For more information about servicing tools, see Servicing tools.

Long-term Servicing Channel

Specialized systems—such as devices that control medical equipment, point-of-sale systems, and ATMs—often require a longer servicing option because of their purpose. These devices typically perform a single important task and don't need feature updates as frequently as other devices in the organization. It's more important that these devices be kept as stable and secure as possible than up to date with user interface changes. The LTSC servicing model prevents Enterprise LTSC devices from receiving the usual feature updates and provides only quality updates to ensure that device security stays up to date. With this in mind, quality updates are still immediately available to Windows 10 Enterprise LTSC clients, but customers can choose to defer them by using one of the servicing tools mentioned in the section Servicing tools.

The Long-term Servicing channel isn't intended for deployment on most or all the devices in an organization. It should be used only for special-purpose devices. As a general guideline, a device with Microsoft Office installed is a general-purpose device, typically used by an information worker, and therefore it's better suited for the General Availability channel.

Microsoft never publishes feature updates through Windows Update on devices that run Windows 10 Enterprise LTSC. Instead, it typically offers new LTSC releases every 2-3 years, and organizations can choose to install them as in-place upgrades or even skip releases over the product's lifecycle. Always check your individual LTSC release to verify its servicing lifecycle. For more information, see release information, or perform a search on the product's lifecycle information page.

> **Note**
>
> LTSC releases will support the currently released processors and chipsets at the time of release of the LTSC. As future CPU generations are released, support will be created through future LTSC releases that customers can deploy for those systems. For more information, see **Supporting the latest processor and chipsets on Windows** in Lifecycle support policy FAQ - Windows Products.

The Long-term Servicing Channel is available only in the Windows Enterprise LTSC editions. This edition of Windows doesn't include some applications, such as Microsoft Edge, Microsoft Store, Microsoft Mail, Calendar, OneNote, Weather, News, Sports, Money, Photos, Camera, Music, and Clock. These apps aren't supported in the Enterprise LTSC editions, even if you install by using sideloading.

Windows Insider

For many IT pros, gaining visibility into feature updates early can be both intriguing and valuable for future end user communications as well as provide the means to test for any issues on the next General Availability release. Windows Insiders can consume and deploy preproduction code to their test machines, gaining early visibility into the next build. Testing the early builds helps both Microsoft and its customers because they have the opportunity to discover possible issues before the update is ever publicly available and can report it to Microsoft.

Microsoft recommends that all organizations have at least a few devices enrolled in the Windows Insider Program and provide feedback on any issues they encounter. For information about the Windows Insider Program for Business, go to Windows Insider Program for Business.

Servicing tools

There are many tools you can use to service Windows as a service. Each option has its pros and cons, ranging from capabilities and control to simplicity and low administrative requirements. The following are examples of the servicing tools available to manage Windows updates:

- **Windows Update (stand-alone)** provides limited control over feature updates, with IT pros manually configuring the device to be in the General Availability Channel. Organizations can target which devices defer updates by selecting the **Defer upgrades** check box in **Start\Settings\Update & Security\Advanced Options** on a Windows client device.
- **Windows Update client policies** include control over update deferment and provides centralized management using Group Policy or MDM. Windows Update client policies can be used to defer updates by up to 365 days, depending on the version. These deployment options are available to clients in the General Availability Channel. In addition to being able to use Group Policy to manage Windows Update client policies, either option can be configured without requiring any on-premises infrastructure by using Microsoft Intune.
- **Windows Server Update Services (WSUS)** provides extensive control over updates and is natively available in the Windows Server operating system. In addition to the ability to defer updates, organizations can add an approval layer for updates and choose to deploy them to specific computers or groups of computers whenever ready.
- **Microsoft Configuration Manager** provides the greatest control over servicing Windows as a service. IT pros can defer updates, approve them, and have multiple options for targeting deployments and managing bandwidth usage and deployment times.

Servicing tools comparison

Expand table

Servicing tool	Can updates be deferred?	Ability to approve updates	Peer-to-peer option	Additional features
Windows Update	Yes (manual)	No	Delivery Optimization	None
Windows Update client policies	Yes	No	Delivery Optimization	Other Group Policy objects
WSUS	Yes	Yes	BranchCache or Delivery Optimization	Upstream/downstream server scalability
Configuration Manager	Yes	Yes	BranchCache, Client Peer Cache, or Delivery Optimization. For the latter, see peer-to-peer content distribution and Optimize Windows Update Delivery	Distribution points, multiple deployment options

Last updated on 06/17/2025

Update release cycle for Windows clients

Article • 03/27/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

Windows updates help you to stay productive and protected. They provide your users and IT administrators with the security fixes they need, and protect devices so that unpatched vulnerabilities can't be exploited. Updates for the Windows client OS are typically cumulative. They include all previously released fixes to guard against fragmentation of the operating system. Reliability and vulnerability issues can occur when only a subset of fixes is installed.

This article provides details on the types of updates that Microsoft provides, and how they help make the overall user experience simple and consistent.

> **Tip**
>
> If you'd like to know what's coming to Windows, check out the Windows Roadmap. The roadmap is a high-level overview of the features and functionality that are planned for future releases of Windows.

Types of update releases

Expand table

Release type	Description	Release cycle
Monthly security update release	A cumulative update release that includes both security and nonsecurity content	Second Tuesday of each month, typically published at 10:00 AM Pacific Time (PST/PDT)
Optional nonsecurity preview release	An optional cumulative update release that's typically used for early validation of the monthly security update release	Fourth Tuesday of each month, typically published at 10:00 AM Pacific Time (PST/PDT)
Out-of-band (OOB) release	Resolves a recently identified issue or vulnerability	As needed
Annual feature update	An update with new features and enhancements that also changes the Windows version	Once a year in the second half of the calendar year

Release type	Description	Release cycle
Continuous innovation for Windows 11	Introduces new features and enhancements for Windows 11	Periodically included in an optional nonsecurity preview release then in the monthly security update releases

Monthly security update release

Most people are familiar with the **monthly security update release**. The **monthly security update release** is published on the second Tuesday of each month, typically at 10:00 AM Pacific Time (PST/PDT). This release might commonly be referred to as:

- Patch Tuesday
- Update Tuesday
- B week releases (meaning the second week of the month)
- Quality updates
- Security updates
- Latest cumulative update (LCU)

Monthly security update releases are cumulative. The release includes both new and previously released security fixes, along with nonsecurity content introduced in the prior month's **Optional nonsecurity preview release**. These updates help keep Windows devices secure and compliant by deploying stability fixes and addressing security vulnerabilities. Most organizations consider monthly security update releases as mandatory.

Monthly security update releases are available through the following channels:

- Windows Update
- Windows Server Update Services (WSUS)
- The Microsoft Update Catalog

Many update management tools, such as Microsoft Configuration Manager and Microsoft Intune, rely on these channels for update deployment.

Starting Windows 11, version 24H2, Microsoft may periodically release cumulative updates as checkpoints. The subsequent updates will consist of:

- The update package files associated with the checkpoints, and
- New update package files that contain incremental binary differentials against the version of binaries in the last checkpoint.

Multiple checkpoints may be shipped during the lifecycle of a given Windows release. Devices updating from Windows Update and WSUS can continue to seamlessly install the latest monthly security update regardless of whether there are any preceding checkpoint cumulative updates, **no change is needed to their update process**. Catalog users can review Checkpoint cumulative updates and Microsoft Update Catalog usage for reference.

Optional nonsecurity preview release

Optional nonsecurity preview releases provide IT admins an opportunity for early validation of that content prior to the **monthly security update release**. Admins can test and validate production-quality releases ahead of the planned monthly security update release for the following month. These updates are optional, cumulative, nonsecurity preview releases. New features might initially be deployed in the prior month's **optional nonsecurity preview release**, then ship in the following **monthly security update release**. **Optional nonsecurity preview releases** are typically released on the fourth Tuesday of the month at 10:00 AM Pacific Time (PST/PDT). These releases are only offered to the most recent, supported versions of Windows.

Optional nonsecurity preview releases might commonly be referred to as:

- D week releases (meaning the fourth week of the month)
- Preview updates
- Preview CU
- LCU preview

To access the optional nonsecurity preview release:

- Navigate to **Settings** > **Update & Security** > **Windows Update** and select **Check for updates**.
- Use Windows Insider Program for Business
- Use the Microsoft Update Catalog.

Starting Windows 11, version 24H2, Microsoft may periodically release cumulative updates as checkpoints. The subsequent updates will consist of:

- The update package files associated with the checkpoints, and
- New update package files that contain incremental binary differentials against the version of binaries in the last checkpoint.

Multiple checkpoints may be shipped during the lifecycle of a given Windows release. Devices updating from Windows Update and WSUS can continue to seamlessly install the latest monthly security update regardless of whether there are any preceding

checkpoint cumulative updates, **no change is needed to their update process**. Catalog users can review Checkpoint cumulative updates and Microsoft Update Catalog usage for reference.

OOB releases

Out-of-band (OOB) releases might be provided to fix a recently identified issue or vulnerability. They're used in atypical cases when an issue is detected and can't wait for the next monthly release, because devices must be updated immediately to address security vulnerabilities or to resolve a quality issue impacting many devices. **Out-of-band (OOB) releases** are provided outside of the monthly schedule when there's an exceptional need.

Some key considerations about OOB releases include:

- OOB releases are always cumulative.
  - OOB releases supersede any prior monthly security update and optional nonsecurity preview release.
- OOB releases generally require IT admins to deploy off-cycle.
- Some OOB releases are classified as critical.
  - Critical OOB releases are automatically available to WSUS and Windows Update client policies, just like the monthly security update releases.
- Some OOB releases are classified as noncritical.
  - Noncritical releases only go to the Microsoft Update Catalog for users or organizations to voluntarily obtain the update.

Starting Windows 11, version 24H2, Microsoft may periodically release cumulative updates as checkpoints. The subsequent updates will consist of:

- The update package files associated with the checkpoints, and
- New update package files that contain incremental binary differentials against the version of binaries in the last checkpoint.

Multiple checkpoints may be shipped during the lifecycle of a given Windows release. Devices updating from Windows Update and WSUS can continue to seamlessly install the latest monthly security update regardless of whether there are any preceding checkpoint cumulative updates, **no change is needed to their update process**. Catalog users can review Checkpoint cumulative updates and Microsoft Update Catalog usage for reference.

Continuous innovation for Windows 11

Starting with Windows 11, version 22H2, new features and enhancements are introduced periodically to provide continuous innovation for Windows 11. These features and enhancements use the normal update servicing channels you're already familiar with. At first, new features are introduced with an **optional nonsecurity preview release** and gradually rolled out to unmanaged clients. These new features are released later as part of a **monthly security update release**.

Some of the new features may be disruptive to organizations. By default, these select features are turned off temporarily for all managed devices until the next annual feature update is installed. In this scenario, a device is considered managed if it uses one of the following to determine which updates to install:

- Windows Update client policies
  - Devices that have updates managed by Microsoft Intune use Windows Update client policies
- WSUS
  - Devices that have updates managed by Configuration Manager use WSUS

Features that are turned off by default are listed in the KB article for the monthly cumulative update. If you want to enable these features, there's a client policy that allows admins to enable features that are behind temporary enterprise control. For more information about this policy, see Enable features that are behind temporary enterprise feature control.

Annual feature updates

Annual feature updates are released in the second half of the calendar year. These updates are typically cumulative and include all previously released fixes. They also include new features and enhancements. The annual feature update marks the start of the support lifecycle:

- 24 months of support for Home and Pro editions of Windows
- 36 months of support for Enterprise and Education editions

For more information, see Windows lifecycle FAQ.

Release information

For more information about specific releases, see:

- Windows 11, version 22H2 update history
- Windows 11, version 21H2 update history

- Windows 10 update history
- Windows release health
- What's new in Windows

Feedback

Was this page helpful? Yes No

Provide product feedback

Windows client updates, channels, and tools

Article • 08/23/2023 • Applies to: ✅ Windows 11, ✅ Windows 10

This article provides a brief summary of the kinds of Windows updates, the channels they're served through, and the tools for managing them.

How Windows updates work

There are four phases to the Windows update process:

- **Scan:** A device checks the Microsoft Update server, Windows Update service, or your Windows Server Update Services (WSUS) endpoint at random intervals to see if any updates have been added since the last time updates were searched, and then evaluates whether the update is appropriate by checking the policies that have been set up by the administrator. This process is invisible to the user.
- **Download:** Once the device determines that an update is available, it begins downloading the update. The download process is also invisible to the user. With feature updates, download happens in multiple sequential phases.
- **Install:** After the update is downloaded, depending on the device's Windows Update settings, the update is installed on the system.
- **Commit and restart:** Once installed, the device usually (but not always) must be restarted in order to complete the installation and begin using the update. Before that happens, a device is still running the previous version of the software.

Types of updates

We include information here about many different update types you hear about, but the two overarching types that you have the most direct control over are *feature updates* and *quality updates*.

- **Feature updates:** Released annually. Feature updates add new features and functionality to Windows 10. Because they're delivered frequently (rather than every 3-5 years), they're easier to manage.
- **Quality updates:** Quality updates deliver both security and nonsecurity fixes. Quality updates include security updates, critical updates, servicing stack updates, and driver updates. They're typically released on the second Tuesday of each month, though they can be released at any time. The second-Tuesday releases are the ones that focus on security updates. Quality updates are *cumulative*, so

installing the latest quality update is sufficient to get all the available fixes for a specific feature update, including any out-of-band security fixes and any *servicing stack updates* that might have been released previously.

- **Servicing stack updates:** The "servicing stack" is the code component that actually installs Windows updates. From time to time, the servicing stack itself needs to be updated in order to function smoothly. If you don't install the latest servicing stack update, there's a risk that your device can't be updated with the latest Microsoft security fixes. Servicing stack updates aren't necessarily included in *every* monthly quality update, and occasionally are released out of band to address a late-breaking issue. Always install the latest available quality update to catch any servicing stack updates that might have been released. The servicing stack also contains the "component-based servicing stack" (CBS), which is a key underlying component for several elements of Windows deployment, such as DISM, SFC, changing Windows features or roles, and repairing components. The CBS is a small component that typically doesn't have updates released every month. You can find a list of servicing stack updates at Latest servicing stack updates. For more detail about servicing stack updates, see Servicing stack updates.
- **Driver updates**: These update drivers applicable to your devices. Driver updates are turned off by default in Windows Server Update Services (WSUS), but for cloud-based update methods, you can control whether they're installed or not.
- **Microsoft product updates:** These update other Microsoft products, such as Office. You can enable or disable Microsoft updates by using policies controlled by various servicing tools.

Servicing channels

There are three servicing channels, each of which offers you a different level of flexibility with how and when updates are delivered to devices. Using the different servicing channels allows you to deploy Windows "as a service," which conceives of deployment as a continual process of updates that roll out across the organization in waves. In this approach, an update is plugged into this process and while it runs, you monitor for anomalies, errors, or user impact and respond as issues arise--without interrupting the entire process.

The first step of controlling when and how devices install updates is assigning them to the appropriate servicing channel. You can assign devices to a particular channel with any of several tools, including Microsoft Configuration Manager, Windows Server Update Services (WSUS), and Group Policy settings applied by any of several means. By dividing devices into different populations ("deployment groups" or "rings") you can use servicing channel assignment, followed by other management features such as update deferral policies, to create a phased deployment of any update that allows you to start

with a limited pilot deployment for testing before moving to a broad deployment throughout your organization.

General Availability Channel

In the General Availability Channel, feature updates are released annually. As long as a device isn't set to defer feature updates, any device in this channel installs a feature update as soon as it's released. If you use Windows Update client policies, the channel provides three months of additional total deployment time before being required to update to the next release.

Windows Insider Program for Business

Insider preview releases are made available during the development of the features that will be shipped in the next feature update, enabling organizations to validate new features and compatibility with existing apps and infrastructure, providing feedback to Microsoft on any issues encountered. There are options within the Windows Insider Program for Business channel:

- Windows Insider Canary
- Windows Insider Dev
- Windows Insider Beta
- Windows Insider Release Preview

We recommend that you use the Windows Insider Release Preview channel for validation activities.

Long-term Servicing Channel

The **Long-Term Servicing Channel** is designed to be used only for specialized devices (which typically don't run Office) such as ones that control medical equipment or ATMs. Devices on this channel receive new feature releases every two to three years. LTSC releases service a special LTSC edition of Windows 10 and are only available through the Microsoft Volume Licensing Center.

The General Availability Channel is the default servicing channel for all Windows devices except those with the LTSC edition installed. The following table shows the servicing channels available to each edition.

Expand table

Edition	General Availability Channel	Insider Program	Long-Term Servicing Channel
Home	Yes	No	No
Pro	Yes	Yes	No
Enterprise	Yes	Yes	No
Enterprise LTSC	No	No	Yes
Pro Education	Yes	Yes	No
Education	Yes	Yes	No

Servicing tools

Tools for on-premises update delivery

Windows Server Update Services (WSUS): you set up a WSUS server, which downloads updates in bulk from Microsoft. Your individual devices then connect to your server to install their updates from there.

You can set up, control, and manage the server and update process with several tools:

- A standalone Windows Server Update Services server operated directly
- Configuration Manager
- Non-Microsoft tools

For more information, see Windows Server Update Services (WSUS).

Tools for cloud-based update delivery

Your individual devices connect to Microsoft endpoints directly to get the updates. The details of this process (how often devices download updates of various kinds, from which channels, deferrals, and details of the users' experience of installation) are set on devices either with Group Policy or MDM policies, which you can control with any of several tools:

- Group Policy Management Console (Gpmc.msc)
- Microsoft Intune
- Non-Microsoft MDM tools

Hybrid scenarios

It's also possible to combine WSUS-based on-premises update distribution with cloud-based update delivery.

Feedback

Was this page helpful? Yes No

Provide product feedback

Defining Windows update-managed devices

Article • 06/25/2024 • Applies to: ✅ Windows 11, ✅ Windows 10

As an IT administrator, understanding the differences between managed and unmanaged devices is crucial for effective Windows update management. This article provides clarity on the terminology and practices involved in managing Windows updates for both types of devices.

What are update-managed Windows devices?

Update-managed devices are those where an IT administrator or organization controls Windows updates through a management tool, such as Microsoft Intune, or by directly setting policies. You can directly set policies with group policy objects (GPO), configuration service provider (CSP) policies, or Microsoft Graph.

> **Note**
>
> This definition is true even if you directly set registry keys. However, we don't recommended doing this action because registry keys can be easily overwritten.

Managed devices can include desktops, laptops, tablets, servers, and manufacturing equipment. These devices are secured and configured according to your organization's standards and policies.

IT-managed: Windows update offering

Devices are considered Windows update-managed if you manage the update offering in the following ways:

- You configure policies to manage which updates are offered to the specific device.
- You set when your organization should receive feature, quality, and driver updates, among others.
- You use group policy objects (GPO), configuration service provider (CSP), or Microsoft Graph to configure these offerings.

IT-managed: Windows update experience

Devices are considered Windows update-managed if you use policies (GPO, CSP, or Microsoft Graph) to manage device behavior when taking Windows updates.

Examples of controllable device behavior include active hours, update grace periods and deadlines, update notifications, update scheduling, and more. Consult the complete list at Update Policy CSP.

Examples of update-managed Windows devices

Here are a few examples of update-managed devices:

- **Company-owned devices:** Devices provisioned by your IT department with corporate credentials, configurations, and policies.
- **Employee-owned devices in BYOD programs:** Personally owned devices that are enrolled in the company's device management system to securely access corporate resources.
- **Devices provisioned through Windows Autopilot:** Devices that are set up and preconfigured to be business-ready right out of the box.
- **Mandated security settings:** Devices with health requirements such as device encryption, PIN or strong password, specific inactivity timeout periods, and up-to-date operating systems.
- **Intune-enrolled devices:** Devices enrolled in Microsoft Intune for network access and enforced security policies.
- **Third-party managed devices:** Devices enrolled in non-Microsoft management tools with configured Windows update policies via GPO, CSP, or registry key.

What are update-unmanaged Windows devices?

Unlike update-managed devices, unmanaged devices aren't controlled through policies, management tools, or software. These devices aren't enrolled in tools like Microsoft Intune or Configuration Manager. If you only configure the Settings page to control overall device behavior when taking updates, it's considered an unmanaged device.

> **Note**

The term "Microsoft managed devices" used to refer to what we now call "update unmanaged Windows devices." Based on feedback, we have updated our terminology for clarity.

Examples of update-unmanaged Windows devices

Examples of update-unmanaged devices include:

- **Personal devices:** Devices owned by individuals at your organization that aren't enrolled in any corporate management system.
- **BYOD devices not enrolled in management programs:** Devices used for work but not part of an organizational bring your own device (BYOD) program.
- **Peripheral devices:** Devices like printers, IP phones, and uninterruptible power supplies (UPS) that can't accept centrally managed administrative credentials.

For more information on managed and unmanaged devices, see Secure managed and unmanaged devices.

Feedback

Was this page helpful? Yes No

Provide product feedback

Prepare a servicing strategy for Windows client updates

Article • 08/24/2023 • Applies to: ✅ Windows 11, ✅ Windows 10

> **Looking for consumer information?** See Windows Update: FAQ

Here's an example of what this process might look like:

- **Configure test devices.** Configure test devices in the Windows Insider Program so that Insiders can test feature updates before they're available to the General Availability Channel. Typically, this population would be a few test devices that IT staff members use to evaluate prerelease builds of Windows. Microsoft provides current development builds to Windows Insider members approximately every week so that interested users can see the functionality Microsoft is adding. See the section Windows Insider for details on how to enroll in the Windows Insider Program for Business.
- **Identify excluded devices.** For some organizations, special-purpose devices, like devices that control factory or medical equipment or run ATMs, require a stricter, less frequent feature update cycle than the General Availability Channel can offer. For those devices, install the Enterprise LTSC edition to avoid feature updates for up to 10 years. Identify these devices, and separate them from the phased deployment and servicing cycles to help remove confusion for your administrators and ensure that devices are handled correctly.
- **Recruit volunteers.** The purpose of testing a deployment is to receive feedback. One effective way to recruit pilot users is to request volunteers. When doing so, clearly state that you're looking for feedback rather than people to just "try it out" and that there could be occasional issues involved with accepting feature updates right away. With Windows as a service, the expectation is that there should be few issues, but if an issue does arise, you want testers to let you know as soon as possible. When considering whom to recruit for pilot groups, be sure to include members who provide the broadest set of applications and devices to validate the largest number of apps and devices possible.
- **Update Group Policy.** Each feature update includes new group policies to manage new features. If you use Group Policy to manage devices, the Group Policy Admin for the Active Directory domain needs to download an .admx package and copy it to their Central Store (or to the PolicyDefinitions directory in the SYSVOL folder of a domain controller if not using a Central Store). You can manage new group policies from the latest release of Windows by using Remote Server Administration Tools. The ADMX download package is created at the end of each development

cycle and then posted for download. To find the ADMX download package for a given Windows build, search for "ADMX download for Windows build xxxx". For details about Group Policy management, see How to create and manage the Central Store for Group Policy Administrative Templates in Windows

- **Choose a servicing tool.** Decide which product you'll use to manage the Windows updates in your environment. If you're currently using Windows Server Update Services (WSUS) or Microsoft Configuration Manager to manage your Windows updates, you can continue using those products to manage Windows 10 or Windows 11 updates. Alternatively, you can use Windows Update client policies. In addition to which product you'll use, consider how you'll deliver the updates. Multiple peer-to-peer options are available to make update distribution faster. For a comparison of tools, see Servicing tools.
- **Prioritize applications.** First, create an application portfolio. This list should include everything installed in your organization and any webpages your organization hosts. Next, prioritize this list to identify those apps that are the most business critical. Because the expectation is that application compatibility with new versions of Windows will be high, only the most business-critical applications should be tested before the pilot phase; everything else can be tested afterwards. For more information about identifying compatibility issues withe applications, see Manage Windows upgrades with Upgrade Analytics.

Each time Microsoft releases a feature update, the IT department should use the following high-level process to help ensure that the broad deployment is successful:

1. **Validate compatibility of business critical apps.** Test your most important business-critical applications for compatibility with the new Windows 10 feature update running on your Windows Insider machines identified in the earlier "Configure test devices" step of the previous section. The list of applications involved in this validation process should be small because most applications can be tested during the pilot phase.
2. **Target and react to feedback.** Microsoft expects application and device compatibility to be high, but it's still important to have targeted groups within both the IT department and business units to verify application compatibility for the remaining applications in your application portfolio. Because only the most business-critical applications are tested beforehand, this activity represents most of the application compatibility testing in your environment. It shouldn't necessarily be a formal process but rather user validation by using a particular application. So, the next step is to deploy the feature update to early-adopting IT users and your targeted groups running in the General Availability Channel that you identified in the "Recruit volunteers" step of the previous section. Be sure to communicate clearly that you're looking for feedback as soon as possible, and state exactly how

users can submit feedback to you. Should an issue arise, have a remediation plan to address it.

3. **Deploy broadly.** Finally, focus on the large-scale deployment using deployment rings. Build deployment rings that target groups of computers in your selected update-management product. To reduce risk as much as possible, construct your deployment rings in a way that splits individual departments into multiple rings. This way, if you were to encounter an issue, you don't prevent any critical business from continuing. By using this method, each deployment ring reduces risk as more people have been updated in any particular department.

Feedback

Was this page helpful? Yes No

Provide product feedback

Plan for Windows 11

Article • 03/27/2025 • Applies to: ✅ Windows 11

This article provides guidance to help you plan for Windows 11 in your organization.

Deployment planning

Since Windows 11 is built on the same foundation as Windows 10, you can use the same deployment capabilities, scenarios, and tools. You can also use the same basic deployment strategy that you use today for Windows 10. Make sure that you review and update your servicing strategy to adjust for changes in servicing and support for Windows 11.

At a high level, this strategy should include the following steps:

- Create a deployment plan
- Define readiness criteria
- Evaluate infrastructure and tools
- Test applications
- Define your servicing strategy

If you're looking for ways to optimize your approach to deploying Windows 11, or if deploying a new version of Windows isn't a familiar process for you, consider the factors in the following sections.

Determine eligibility

As a first step, determine which of your current devices meet the Windows 11 hardware requirements. To ensure compatibility, verify that your device meets or exceeds Windows 11 requirements.

Microsoft has analysis tools to help you evaluate your devices against the Windows 11 hardware requirements. If you're running Windows 10 Home, Pro, or Pro for Workstations editions, you can use the PC Health Check app to determine Windows 11 eligibility. Users running Windows 10 Enterprise and Education editions should rely on their IT administrators to let them know when they're eligible for the upgrade.

Enterprise organizations looking to evaluate device readiness in their environments can expect this capability to be integrated into existing Microsoft tools, such as Endpoint analytics.

Windows 11 availability

The availability of Windows 11 varies according to a device's hardware and whether the device receives updates directly from Microsoft, or from a management solution that's maintained by an IT administrator.

For more information, see Defining Windows update-managed devices.

Managed devices

Managed devices are devices that are under organizational control. Managed devices include those devices managed by Microsoft Intune, Microsoft Configuration Manager, or other endpoint management solutions.

If you manage devices on behalf of your organization, you can upgrade eligible devices to Windows 11 using your existing deployment and management tools.

Organizations that use Windows Update client policies also have the following benefits:

- Ensuring that devices that don't meet the minimum hardware requirements aren't automatically offered the Windows 11 upgrade.
- More insight into safeguard holds. While safeguard holds function for Windows 11 devices just as they do for Windows 10 today, administrators using Windows Update client policies have access to information on which safeguard holds are preventing individual devices from taking the upgrade to Windows 11.

> **Note**
>
> Also, Windows 11 has new Microsoft Software License Terms. If you deploy with Windows Update client policies or Windows Server Update Services, you accept these new license terms on behalf of the users in your organization.

Unmanaged devices

Unmanaged devices are devices that an IT administrator doesn't manage on behalf of an organization. For OS deployment, these devices aren't subject to organizational policies that manage upgrades or updates.

Windows 11 was offered to eligible Windows 10 devices in 2021. Messaging on new devices varies by PC manufacturer.

The Windows 11 upgrade is available on eligible, unmanaged devices to users who manually seek the upgrade through Windows Update. As with all Windows Update-managed devices, the **Windows Update** settings page confirms when a device is eligible.

Just like Windows 10, the machine learning-based intelligent rollout process is used when rolling out upgrades. Machine learning uses a combination of testing, close partner engagement, feedback, diagnostic data, and real-life insights to manage quality. This process improves the update experience and ensures that devices first nominated for updates are the devices likely to have a seamless experience. Devices that might have compatibility issues with the upgrade get the benefit of resolving these issues before the upgrade is offered.

Windows 11 readiness considerations

The recommended method to determine if your infrastructure, deployment processes, and management tools are ready for Windows 11 is to join the Windows Insider Program for Business. As a participant in the release preview channel, you can validate that your devices and applications work as expected, and explore new features.

As you plan your endpoint management strategy for Windows 11, consider moving to cloud-based mobile device management (MDM), such as Microsoft Intune. If a cloud-only approach isn't right for your organization yet, you can still modernize and streamline essential pieces of your endpoint management strategy as follows:

- To manage Configuration Manager clients over the internet, create a cloud management gateway (CMG).
- Attach your existing Configuration Management estate to the cloud with tenant attach so you can manage all devices from within the Microsoft Intune admin center.
- Use co-management to concurrently manage devices using both Configuration Manager and Microsoft Intune. This concurrent management allows you to take advantage of cloud-powered capabilities like conditional access.

For more information on the benefits of these approaches, see Cloud Attach Your Future: The Big 3.

The introduction of Windows 11 is also a good time to review your hardware refresh plans and prioritize eligible devices to ensure an optimal experience for your users.

Servicing and support

Along with user experience and security improvements, Windows 11 introduces enhancements to Microsoft's servicing approach based on your suggestions and feedback.

- **Quality updates**: Windows 11 and Windows 10 devices receive regular monthly quality updates to provide security updates and bug fixes.
- **Feature updates**: Microsoft provides a single Windows 11 feature update annually, targeted for release in the second half of each calendar year.
- **Lifecycle**:
  - Home, Pro, Pro for Workstations, and Pro for Education editions of Windows 11 receive 24 months of support from the general availability date.
  - Enterprise and Education editions of Windows 11 are supported for 36 months from the general availability date.

A consolidated Windows 11 update history is available. Similarly, the Windows release health hub offers quick access to Windows 11 servicing announcements, known issues, and safeguard holds.

It's important that organizations have adequate time to plan for Windows 11. Microsoft also recognizes that many organizations have a mix of Windows 11 and Windows 10 devices across their ecosystem. Devices on in-service versions of Windows 10 continue to receive monthly Windows 10 security updates, and incremental improvements to Windows 10 to support ongoing Microsoft 365 deployments. For more information, see the Windows 10 release information page.

> **Tip**
>
> If you'd like to know what's coming to Windows, check out the **Windows Roadmap**. The roadmap is a high-level overview of the features and functionality that are planned for future releases of Windows.

Application compatibility

Microsoft's compatibility promise for Windows 10 is maintained for Windows 11. Data from the App Assure program shows that Windows 10 compatibility rates are over 99.7% for enterprise organizations, including line of business (LOB) apps. Microsoft remains committed to ensuring that the apps you rely upon continue to work as expected when you upgrade. Windows 11 is subject to the same app compatibility

validation requirements that are in place for Windows 10 today, for both feature and quality updates.

For more information, see Windows compatibility cookbook.

App Assure

If you run into compatibility issues or want to ensure that your organization's applications are compatible from day one, App Assure can help. With enrollment in the App Assure service, any app compatibility issues that you find with Windows 11 can be resolved. Microsoft helps you remedy application issues at no cost. Since 2018, App Assure has evaluated almost 800,000 apps, and subscriptions are free for eligible customers with more than 150 devices.

Next steps

Prepare for Windows 11

Plan to deploy updates for Windows 10 and Microsoft 365 Apps

Feedback

Was this page helpful? Yes No

Provide product feedback

Create a deployment plan

Article • 02/23/2023 • Applies to: ✅ Windows 11, ✅ Windows 10

A "service management" mindset means that the devices in your organization fall into a continuum, with the software update process being constantly planned, deployed, monitored, and optimized. Once you use this process for feature updates, quality updates become a lightweight procedure that is simple and fast to execute, ultimately increasing velocity.

When you move to a service management model, you need effective ways of rolling out updates to representative groups of devices. We've found that a ring-based deployment works well for us at Microsoft and many other organizations across the globe. Deployment rings in Windows clients are similar to the deployment groups most organizations constructed for previous major revision upgrades. They're simply a method to separate devices into a deployment timeline.

At the highest level, each ring comprises a group of users or devices that receive a particular update concurrently. For each ring, IT administrators set criteria to control deferral time or adoption (completion) that should be met before deployment to the next broader ring of devices or users can occur.

A common ring structure uses three deployment groups:

- Preview: Planning and development
- Limited: Pilot and validation
- Broad: Wide deployment

> **Note**
>
> Organizations often use different names for their rings, for example:
>
> - First > Fast > Broad
> - Canaries > Early Adopters > Users
> - Preview > Broad > Critical

How many rings should I have?

There are no definite rules for exactly how many rings to have for your deployments. As mentioned previously, you might want to ensure zero downtime for mission-critical devices by putting them in their own ring. If you have a large organization, you might

want to consider assigning devices to rings based on geographic location. Or assign based on the size of rings so that helpdesk resources are more available. Consider the needs of your business and introduce rings that make sense for your organization.

Advancing between rings

There are basically two strategies for moving deployments from one ring to the next. One is service-based, the other project-based.

- "Red button" (service-based): Assumes that content is good until proven bad. Content flows until an issue is discovered, at which point the IT administrator presses the "red button" to stop further distribution.
- "Green button" (project-based): Assumes that content is bad until proven good. Once all validation has passed, the IT administrator presses the "green button" to push the content to the next ring.

When it comes to deployments, having manual steps in the process usually impedes update velocity. A "red button" strategy is better when that is your goal.

Preview ring

The purpose of the Preview ring is to evaluate the new features of the update. It's *not* for broad parts of the organization but is limited to the people who are responsible for knowing what is coming next, generally IT administrators. Ultimately, this phase is the time the design and planning work happens so that when the public update is shipped, you can have greater confidence in the update.

> **Note**
>
> Being part of the Windows Insider Program gives you early access to Windows releases so that you can use Insider Preview builds in your Preview ring to validate your apps and infrastructure, preparing you for public Windows releases.

Who goes in the Preview ring?

The Preview ring users are the most tech-savvy and resilient people, who won't lose productivity if something goes wrong. In general, these users are IT pros, and perhaps a few people in the business organization.

During your plan and preparation phases, you should focus on the following activities:

- Work with Windows Insider Preview builds.
- Identify the features and functionality your organization can or wants to use.
- Establish who will use the features and how they'll benefit.
- Understand why you're putting out the update.
- Plan for usage feedback.

Remember, you're working with pre-release software in the Preview ring and you'll be evaluating features and testing the update for a targeted release.

> **Important**
>
> If you are using Windows Insider (pre-release) releases for your preview ring and you are using WSUS or Windows Update client policies, be sure to set the following policies to allow for Preview builds:
>
> - **Manage Preview Builds: 2 - Enable preview builds** • Under **Branch Readiness Level**, select **When Preview Builds and Feature Updates are Received: 4--Windows Insider Program Slow**

Limited ring

The purpose of the Limited ring is to validate the update on representative devices across the network. During this period, data, and feedback are generated to enable the decision to move forward to broader deployment. Desktop Analytics can help with defining a good Limited ring of representative devices and help monitor the deployment.

Who goes in the Limited ring?

The most important part of this phase is finding a representative sample of devices and applications across your network. If possible, all hardware and all applications should be represented. It's important that the people selected for this ring are using their devices regularly to generate the data you'll need to make a decision for broader deployment across your organization. The IT department, lab devices, and users with the most cutting-edge hardware usually don't have the applications or device drivers that are truly a representative sample of your network.

During your pilot and validation phases, you should focus on the following activities:

- Deploy new innovations.
- Assess and act if issues are encountered.

- Move forward unless blocked.

When you deploy to the Limited ring, you'll be able to gather data and react to incidents happening in the environment, quickly addressing any issues that might arise. Ensure you monitor for sufficient adoption within this ring. Your Limited ring represents your organization across the board. When you achieve sufficient adoption, you can have confidence that your broader deployment will run more smoothly.

Broad deployment

Once the devices in the Limited ring have had a sufficient stabilization period, it's time for broad deployment across the network.

Who goes in the Broad deployment ring?

In most businesses, the Broad ring includes the rest of your organization. Because of the work in the previous ring to vet stability and minimize disruption (with diagnostic data to support your decision), a broad deployment can occur relatively quickly.

> **Note**
>
> In some instances, you might hold back on mission-critical devices (such as medical devices) until deployment in the Broad ring is complete. Get best practices and recommendations for deploying Windows client feature updates to mission-critical devices.

During the broad deployment phase, you should focus on the following activities:

- Deploy to all devices in the organization.
- Work through any final unusual issues that weren't detected in your Limited ring.

Feedback

Was this page helpful? Yes No

Provide product feedback

Define readiness criteria

Article • 01/19/2023 • Applies to: ✅ Windows 11, ✅ Windows 10

Planning and managing a deployment involves a variety of distinct activities and roles best suited to each activity. This article describes how to identify important roles and figure out how to classify apps.

Figure out roles and personnel

As you plan, it's worth figuring out which roles you'll need to carry out the deployment and who should fill them. Different roles are active at various phases of a deployment. Depending on the size and complexity of your organization, some of the roles could be filled by the same person. However, it's best to have an established *process manager*, who will oversee all of the tasks for the deployment.

Process manager

The process manager leads the update deployment process and has the authority to push the process forward--or halt it if necessary. They also have responsibilities in organizing these activities:

Expand table

Compatibility workstream	Deployment	Capability and modernization
Assigning application priority	Reviewing infrastructure requirements	Determining infrastructure changes
Application assessment	Validating infrastructure against requirements	Determining configuration changes
Device assessment	Creating infrastructure update plan	Create capability proposal

It's the process manager's role to collect reports on remediation efforts, escalate failures, and to decide whether your environment is ready for pilot deployment and then broad deployment.

This table sketches out one view of the other roles, with their responsibilities, relevant skills, and the deployment phases where they are needed:

Expand table

Role	Responsibilities	Skills	Active phases
Process manager	Manages the process end to end; ensures inputs and outputs are captures; ensures that activities progress	IT Service Management	Plan, prepare, pilot deployment, broad deployment
Application owner	Define application test plan; assign user acceptance testers; certify the application	Knowledge of critical and important applications	Plan, prepare, pilot deployment
Application developer	Ensure apps are developed to stay compatible with current Windows versions	Application development; application remediation	Plan, prepare
End-user computing	Typically a group including infrastructure engineers or deployment engineers who ensure upgrade tools are compatible with Windows	Bare-metal deployment; infrastructure management; application delivery; update management	Plan, prepare, pilot deployment, broad deployment
Operations	Ensure that support is available for current Windows version. Provide post-deployment support, including user communication and rollbacks.	Platform security	Prepare, pilot deployment, broad deployment
Security	Review and approve the security baseline and tools	Platform security	Prepare, pilot deployment
Stakeholders	Represent groups affected by updates, for example, heads of finance, end-user services, or change management	Key decision maker for a business unit or department	Plan, pilot deployment, broad deployment

Set criteria for rating apps

Some apps in your environment are fundamental to your core business activities. Other apps help workers perform their roles, but aren't critical to your business operations. Before you start inventorying and assessing the apps in your environment, you should establish some criteria for categorizing your apps, and then determine a priority for each. This process will help you understand how best to deploy updates and how to resolve any issues that could arise.

In the Prepare phase, you'll apply the criteria you define now to every app in your organization.

Here's a suggested classification scheme:

Expand table

Classification	Definition
Critical	The most vital applications that handle core business activities and processes. If these applications were not available, the business, or a business unit, couldn't function at all.
Important	Applications that individual staff members need to support their productivity. Downtime here would affect individual users, but would only have a minimal impact on the business.
Not important	There is no impact on the business if these apps are not available for a while.

Once you have classified your applications, you should agree what each classification means to the organization in terms of priority and severity. This activity will help ensure that you can triage problems with the right level of urgency. You should assign each app a time-based priority.

Here's an example priority rating system; the specifics could vary for your organization:

Expand table

Priority	Definition
1	Any issues or risks identified must be investigated and resolved as soon as possible.
2	Start investigating risks and issues within two business days and fix them *during* the current deployment cycle.
3	Start investigating risks and issues within 10 business days. You don't have to fix them all within the current deployment cycle. However, all issues must be fixed by the end of the next deployment cycle.
4	Start investigating risks and issues within 20 business days. You can fix them in the current or any future development cycle.

Related to priority, but distinct, is the concept of severity. You should define a severity ranking as well, based on how you feel a problem with an app should affect the deployment cycle.

Here's an example:

Expand table

Severity	Effect
1	Work stoppage or loss of revenue
2	Productivity loss for a business unit
3	Productivity loss for individual users
4	Minimal impact on users

Example: a large financial corporation

Using the suggested scheme, a financial corporation might classify their apps like this:

Expand table

App	Classification
Credit processing app	Critical
Frontline customer service app	Critical
PDF viewer	Important
Image-processing app	Not important

Further, they might combine this classification with severity and priority rankings like this:

Expand table

Classification	Severity	Priority	Response
Critical	1 or 2	1 or 2	For 1, stop deployment until resolved; for 2, stop deployment for affected devices or users only.
Important	3 or 4	3 or 4	For 3, continue deployment, even for affected devices, as long as there is workaround guidance.
Not important	4	4	Continue deployment for all devices.

Feedback

Was this page helpful? Yes No

Provide product feedback

Evaluate infrastructure and tools

Article • 10/31/2023 • Applies to: ✅ Windows 11, ✅ Windows 10

Before you deploy an update, assess your deployment infrastructure. For example, management systems like Configuration Manager, Microsoft Intune, or similar. Also assess current configurations such as security baselines, administrative templates, and policies that affect updates. Then set some criteria to define your operational readiness.

Infrastructure

Do your deployment tools need updates?

- If you use Configuration Manager, is it on the current branch with the latest release installed? Being on this branch ensures that it supports the next Windows client feature update. Configuration Manager releases are supported for 18 months.
- Using a cloud-based management tool like Microsoft Intune reduces support challenges, since no related products need to be updated.
- If you use a non-Microsoft tool, check with its product support to make sure you're using the current version and that it supports the next Windows client feature update.

Rely on your experiences and data from previous deployments to help you judge how long infrastructure changes take and identify any problems you've encountered.

Device settings

Make sure your security baseline, administrative templates, and policies have the right settings to support your devices once the new Windows client update is installed.

Security baseline

Keep security baselines current to help ensure that your environment is secure and that new security feature in the coming Windows client update are set properly.

- **Microsoft security baselines**: You should implement security baselines from Microsoft. They're included in the Security Compliance Toolkit, along with tools for managing them.
- **Industry- or region-specific baselines**: Your specific industry or region might have particular baselines that you must follow per regulations. Ensure that any new baselines support the version of Windows client you're about to deploy.

Configuration updates

There are several Windows policies that affect when Windows updates are installed, deferral, end-user experience, and many other aspects. For example, policies set by group policy, Intune, or other methods. Check these policies to make sure they're set appropriately.

- **Windows Administrative templates**: Each Windows client feature update has a supporting Administrative template (.admx) file. Group Policy tools use Administrative template files to populate policy settings in the user interface. The templates are available in the Download Center, for example, this one for Windows 11, version 23H2 .

- **Policies for update compliance and end-user experience**: Several settings affect when a device installs updates, whether and for how long a user can defer an update, restart behavior after installation, and many other aspects of update behavior. It's especially important to look for existing policies that are out of date or could conflict with new ones.

Define operational readiness criteria

When you deploy an update, you need to make sure the update isn't introducing new operational issues. If incidents arise, make sure the needed documentation and processes are available. Work with your operations and support team to define acceptable trends and what documents or processes require updating:

- **Call trend**: Define what percentage increase in calls relating to Windows client feature updates are acceptable or can be supported.
- **Incident trend**: Define what percentage of increase in calls asking for support relating to Windows client feature updates are acceptable or can be supported.
- **Support documentation**: Review supporting documentation that requires an update to support new infrastructure tooling or configuration as part of the Windows client feature update.
- **Process changes:** Define and update any processes that will change as a result of the Windows feature update.

Your operations and support staff can help you determine if the appropriate information is being tracked at the moment. If it isn't, work out how to get this information so you can gain the right insight.

Tasks

Finally, you can begin to carry out the work needed to ensure your infrastructure and configuration can support the update. To help you keep track, you can classify the work into the following overarching tasks:

- **Review infrastructure requirements**: Go over the details of requirements to support the update, and ensure they've all been defined.
- **Validate infrastructure against requirements**: Compare your infrastructure against the requirements that you identified for the update.
- **Define infrastructure update plan**: Detail how your infrastructure must change to support the update.
- **Review current support volume**: Understand the current support volume to understand how much of an effect the update has when you deploy it.
- **Identify gaps that require attention**: Identify issues that you'll need to address to successfully deploy the update. For example, will your infrastructure engineer have to research how a new feature that comes with the update might affect the infrastructure?
- **Define operational update plan**: Detail how your operational services and processes must change to support the update.

Feedback

Was this page helpful? Yes No

Provide product feedback

Define update strategy with a calendar

Article • 08/23/2023 • Applies to: ✅ Windows 11, ✅ Windows 10

Traditionally, organizations treated the deployment of operating system updates (especially feature updates) as a discrete project that had a beginning, a middle, and an end. A release was "built" (usually in the form of an image) and then distributed to users and their devices.

Today, more organizations are treating deployment as a continual process of updates that roll out across the organization in waves. In this approach, an update is plugged into this process and while it runs, you monitor for anomalies, errors, or user impact and respond as issues arise--without interrupting the entire process. Microsoft has been evolving its Windows release cycles, update mechanisms, and relevant tools to support this model. For more information about the Windows lifecycle, see Windows lifecycle FAQ.

We encourage you to deploy every available release and maintain a fast cadence for some portion of your environment. We also recognize that you might have a large number of devices, and a need for little or no disruption. The lifecycle cadence lets you allow some portion of your environment to move faster while the majority can move less quickly.

Calendar approaches

You can use a calendar approach for either a faster twice-per-year cadence or an annual cadence. Depending on company size, installing feature updates less often than once annually risks devices going out of service and becoming vulnerable to security threats, because they stop receiving the monthly security updates once a version is out of support.

Annual approach

Here's a calendar showing an example schedule that applies one Windows feature update per calendar year, aligned with Microsoft Configuration Manager and Microsoft 365 Apps release cycles:

This approach provides approximately 12 months of use from each feature update before the next update is due to be installed by aligning to the Windows H2 feature update.

This cadence might be most suitable for you if any of these conditions apply:

- You're just starting your journey with the Windows servicing process. If you're unfamiliar with new processes that support Windows servicing, moving from a project happening once every three to five years to a feature update process can be daunting. This approach gives you time to learn new approaches and tools to reduce effort and cost.
- You want to wait and see how successful other companies are at adopting a Windows feature update.
- You want to go quickly with feature updates, and want the ability to skip a feature update while keeping Windows serviced in case business priorities change.

Feedback

Was this page helpful? Yes No

Provide product feedback

Windows Update security

Applies to: ✅ Windows 11, ✅ Windows 10

The Windows Update (WU) system ensures devices are updated securely. Its end-to-end protection prevents manipulation of protocol exchanges and ensures only approved content is installed. Some protected environments may need to update firewall and proxy rules to ensure that Windows updates can be properly accessed. This article provides an overview of the security features of Windows Update.

Windows Update security overview

The Windows Update system distributes a multitude of content. Some examples of this content include:

- Updates to the Windows operating system
- Microsoft 365 Apps updates (Office updates)
- Hardware drivers
- Antivirus definitions
- Microsoft Store apps

This system is initiated when a user interacts with the Windows Update settings page or when an application makes a call into the WU client service API. These calls may be made at various times by Microsoft applications and different parts of Windows, such as Microsoft 365 Apps, Microsoft Defender, and Plug and Play (PnP).

When such interactions occur, the Windows Update service running on the device will trigger a series of exchanges over the internet with Microsoft's Windows Update servers. The general workflow is:

1. A Windows device makes multiple connections to Windows Update services using HTTPS (HTTP over TLS, TCP port 443).
2. Update metadata is exchanged over these connections and results in a list of updates, apps, drivers, and other updates.
3. The device decides whether and when to download items from the resulting list.

Once the list of downloads has been decided, the actual update binary files are then downloaded. The download is done via the Delivery Optimization component over a mix of standard HTTP calls (TCP port 80) and secure peer-to-peer network calls (TCP port 7680). Which method used is based on the device's configuration/group policies.

When downloading updates using Delivery Optimization's peer-to-peer (P2P) networking, the content is integrity validated upon receipt from each peer. If the requested content is

unavailable on the P2P network, then the Delivery Optimization component will download it using HTTP.

Regardless of which method is used to download the content, the resulting files are then validated through digital signatures and file hashes before they're installed. The validation confirms that the download is what was intended, is verified as authentic, and hasn't been tampered with.

Securing metadata connections

When Windows Update scans for updates, it goes through a series of metadata exchanges between the device and Windows Update servers. This exchange is done using HTTPS (HTTP over TLS). These secured connections are certificate-pinned, ensuring that:

- The TLS connection's server certificate is validated (certificate trust, expiry, revocation, SAN entries, etc.)
- The certificate's issuer is validated as genuine Microsoft Windows Update

The connection fails if the issuer is unexpected, or not a valid Windows Update intermediate certificate. Certificate pinning ensures that the device is connecting to legitimate Microsoft servers and prevents man-in-the-middle attacks.

Since Windows Update TLS connections are certificate-pinned, it's important that TLS proxies pass these connections without interception. The full list of DNS names that require proxy/firewall exceptions can be found in the Windows Update troubleshooting article.

Microsoft doesn't provide IP addresses or IP ranges for these exceptions because they may differ over time as changes are made for purposes such as traffic load balancing.

Expected Windows Update server usage

The Windows Update service's servers are used solely by WU components. There's no expectation that end users will be interacting with these remote endpoints. Therefore, these service endpoints may not resolve as expected in a web browser. A user casually browsing to these endpoints may notice a lack of adherence to the latest web browser expectations such as publicly trusted PKI, certificate transparency logging, or TLS requirements. This behavior is expected and doesn't limit or otherwise impact the safety and security of the Windows Update system.

Users attempting to browse to the service endpoints may see security warnings and even content access failures. Again, this behavior is expected as the service endpoints aren't designed for web browser access or casual user consumption.

Securing content delivery

The process of downloading update binaries is secured at a layer above the transport. Even though content may be downloaded through standard HTTP (TCP port 80), the content goes through a rigorous security validation process.

Downloads are load balanced through Content Delivery Networks (CDN), so using TLS would break their Microsoft chain-of-custody. Because a TLS connection to a caching CDN terminates at the CDN, not Microsoft, TLS certificates aren't Microsoft specific. This means that the WU client can't prove the trustworthiness of the CDN as Microsoft doesn't control CDN TLS certificates. Additionally, a TLS connection to a CDN doesn't prove content hasn't been manipulated within the CDN's caching network. Therefore, TLS doesn't offer any of the security promises to the end-to-end Windows Update workflow that it otherwise provides.

Regardless of how the content is delivered, once it has been downloaded, it's properly validated. Content is validated for trust, integrity, and intention using various techniques such as digital signature validation and file hash checks. This level of content validation provides even more layers of security than TLS alone.

> **Note**
>
> We collaborate with various global partners to cache content locally while enforcing a strict zero-trust policy. Regardless of the partner, we conduct necessary security checks at the client (Windows) layer to ensure maximum security. Our content delivery partners inside of China include Baidu, CDNetworks, Kingsoft, NGAA, and Tencent China and in the rest of the world include Akamai, Fastly, GCore, Qwilt, and Tencent Global. In addition to traditional CDN delivery, Windows Update content can be delivered from Microsoft Connected Cache servers locally deployed by network operators (ISPs, NSPs, or Internet Exchanges) or Infrastructure providers such as Edgevana. For more info on MCC please click here. To enhance our service offering, we are continuously evaluating the best options and therefore the list of providers is subject to change.

Windows Server Update Services (WSUS)

Enterprises using WSUS have a similar workflow. However, the client devices connect to their enterprise's WSUS server instead of over the internet to Microsoft's servers. It's up to the enterprise to decide whether to use HTTP or TLS (HTTPS) connections for the metadata exchange. Microsoft strongly advises using TLS connections and configuring client devices with appropriate TLS certificate pinning configurations for metadata exchange with WSUS. For more information about WSUS TLS certificate-pinning, see:

- Windows IT Pro Blog: Changes to improve security for Windows devices scanning WSUS
- Windows IT Pro Blog: Scan changes and certificates add security for Windows devices using WSUS for updates

When a WSUS server updates its own update catalog, it connects to Microsoft's server sync services and scans for updates. The WSUS server synchronization process is similar to the metadata exchange process for client devices connecting to Windows Update. The WSUS-to-Microsoft connection is over TLS and is verified by Microsoft certificate, similar to the WU client's TLS certificate-pinning.

Last updated on 09/18/2025

Windows upgrade and migration considerations

Applies to: ✅ Windows 11, ✅ Windows 10

Files and application settings can be migrated to new hardware running the Windows operating system, or they can be maintained during an operating system upgrade on the same computer. This article summarizes the Microsoft tools that can be used to move files and settings between installations in addition to special considerations for performing an upgrade or migration.

Upgrade from a previous version of Windows

Earlier versions of Windows can be upgraded to later versions. As part of the upgrade experience, the newer version of Windows can be installed while retaining applications, files, and settings as they were in the previous version of Windows. If a custom installation of Windows is performed instead of an upgrade, applications and settings aren't maintained. Personal files and all Windows files and directories are moved to a **Windows.old** folder. The data can be accessed in the **Windows.old** folder after Windows Setup is complete.

Migrate files and settings

The User State Migration Tool (USMT) is available to transfer settings:

- Between two computers running Windows, also known as a *side-by-side* migration.
- On a single computer upgrading the version of Windows when not using an in-place upgrade, also known as a *wipe-and-load* or *refresh* migration.

USMT only transfers the program settings, not the programs themselves. USMT is an application intended for administrators who are performing large-scale automated deployments, but it can also be used in smaller migrations.

Migrate with the User State Migration Tool

USMT can be used to automate migration during large deployments of the Windows operating system. USMT uses XML files that define migration rules. The migration rules can be configured to control exactly what USMT migrates and how it migrates the items. For example, USMT can migrate:

- User accounts, including which specific accounts to migrate.

- User files, including which specific files to migrate.
- Operating system settings, including which specific operating system settings to migrate.
- Settings for some applications, including which specific application settings to migrate.

USMT can be used for the following scenarios:

- **Side-by-side** - migration where one device is replaced with a different device.
- **Wipe-and-load/refresh** - migration where Windows is upgraded on a single device.

> **Important**
>
> USMT only supports devices that are joined to a local Active Directory domain. USMT doesn't support Microsoft Entra joined devices.

Upgrade and migration considerations

When upgrading or migrating to a new version of Windows, be aware of the following issues and considerations:

Application compatibility

For more information about application compatibility in Windows, see Compatibility for Windows 11.

Multilingual Windows image upgrades

USMT doesn't support cross-language upgrades when performing multilingual Windows upgrades. If upgrading or migrating an operating system with multiple language packs installed, only the system default user interface (UI) language can be upgraded or migrated to. For example, if English is the default but a Spanish language pack is installed, only English can be upgraded or migrated to.

If a single-language Windows image that matches the system default UI language of a multilingual operating system is being used for a *wipe-and-load/refresh* migration, the migration works. However, all of the language packs are removed. The language packs will need to be reinstalled after the upgrade is completed.

Errorhandler.cmd

If using **Errorhandler.cmd** when upgrading from an earlier version of Windows, copy **Errorhandler.cmd** into the `%WINDIR%\Setup\Scripts` directory on the original installation of Windows. Copying **Errorhandler.cmd** into the `%WINDIR%\Setup\Scripts` directory on the original installation of Windows makes sure that if there are errors during the downlevel phase of Windows Setup, the commands in **Errorhandler.cmd** run. For more information, see Run a script if Windows Setup encounters a fatal error (ErrorHandler.cmd).

Related content

- User State Migration Tool (USMT) overview.
- Windows upgrade paths.
- Windows edition upgrade.

Last updated on 04/08/2025

What is Delivery Optimization?

Applies to: ✅ Windows 11, ✅ Windows 10

> **Looking for Group Policy objects?** See Delivery Optimization reference or the master spreadsheet available at the Download Center for Windows 11 or for Windows 10.

Windows updates, upgrades, and applications can contain packages with large files. Downloading and distributing updates can consume quite a bit of network resources on the devices receiving them. Delivery Optimization is a reliable HTTP downloader with a cloud-managed solution that allows Windows devices to download those packages from alternate sources if desired (such as other devices on the network and/or a dedicated cache server) in addition to the traditional internet-based servers (referred to as 'HTTP sources' throughout Delivery Optimization documents). You can use Delivery Optimization to reduce bandwidth consumption by sharing the work of downloading these packages among multiple devices in your deployment however, the use of peer-to-peer is optional.

To use either the peer-to-peer functionality or the Microsoft Connected Cache features, devices must have access to the Internet and Delivery Optimization cloud services. When Delivery Optimization is configured to use peers and Microsoft Connected Cache, to achieve the best possible content delivery experience, the client connects to Connected Cache and peers in parallel. If the desired content can't be obtained from Connected Cache or peers, Delivery Optimization seamlessly falls back to the HTTP source to get the requested content.

You can use Delivery Optimization with Windows Update, Windows Server Update Services (WSUS), Microsoft Intune/Windows Update client policies, or Microsoft Configuration Manager (when installation of Express Updates is enabled).

For information about setting up Delivery Optimization, including tips for the best settings in different scenarios, see Set up Delivery Optimization. For a comprehensive list of all Delivery Optimization settings, see Delivery Optimization reference.

> **Note**
>
> WSUS can also use BranchCache for content sharing and caching. If Delivery Optimization is enabled on devices that use BranchCache, Delivery Optimization will be used instead.

Requirements

The following table lists the minimum Windows 10 version that supports Delivery Optimization:

Expand table

Device type	Minimum Windows version
Computers running Windows 10 or later	Windows 10 1511
Windows Server 2019 or later (Server Core included)	Windows Server 2019
Windows IoT devices	Windows 10 1803

Types of download content supported by Delivery Optimization

Windows Client

Expand table

Windows Client	Minimum Windows version	HTTP Downloader	Peer to Peer	Microsoft Connected Cache
Windows Update (feature updates quality updates, language packs, drivers)	Windows 10 1511, Windows 11	✔	✔	✔
Windows 10/11 UWP Store apps	Windows 10 1511, Windows 11	✔	✔	✔
Windows 11 Win32 Store apps	Windows 11	✔		
Windows Defender definition updates	Windows 10 1511, Windows 11	✔	✔	✔
Intune Win32 apps	Windows 10 1709, Windows 11	✔ (excluding Government cloud environments)	✔ (excluding Government cloud environments)	✔ (excluding Government cloud environments)
Microsoft 365 Apps and updates	Windows 10 1709, Windows 11	✔	✔ (excluding SAC Extended channel)	✔

Windows Client	Minimum Windows version	HTTP Downloader	Peer to Peer	Microsoft Connected Cache
Edge Browser Updates	Windows 10 1809, Windows 11	✔	✔	✔
Configuration Manager Express updates	Windows 10 1709 + Configuration Manager version 1711, Windows 11	✔	✔	✔
Dynamic updates	Windows 10 1903, Windows 11	✔	✔	✔
MDM Agent	Windows 11	✔		
Xbox Game Pass (PC)	Windows 10 1809, Windows 11	✔		✔
Windows Package Manager	Windows 10 1809, Windows 11	✔		
MSIX Installer	Windows 10 2004, Windows 11	✔		
Teams updates	Windows 10 2004, Windows 11	✔	✔ (starting with version 25122.1415.3698.6812)	✔ (only over HTTPS)

Windows Server

Expand table

Windows Server	Minimum Windows version	HTTP Downloader	Peer to Peer	Microsoft Connected Cache
Windows Update	Windows Server 2019 (1809)	✔	✔	✔
Edge Browser Updates	Windows Server 2019 (1809)	✔	✔	✔

Linux (Public Preview)

Expand table

Linux (Public Preview)	Linux versions	HTTP Downloader	Peer to Peer	Microsoft Connected Cache
Device Update for IoT Hub	Ubuntu 18.04, 20.04 / Debian 9, 10	✔		✔

> **Note**
>
> Starting with Configuration Manager version 1910, you can use Delivery Optimization for the distribution of all Windows update content for clients running Windows 10 version 1709 or newer, not just express installation files. For more, see Delivery Optimization starting in version 1910.

In Windows client Enterprise, Professional, and Education editions, Delivery Optimization is enabled by default for peer-to-peer sharing on the local network (NAT). Specifically, all of the devices must be behind the same NAT (which includes either Ethernet or WiFi), but you can configure it differently in Group Policy and mobile device management (MDM) solutions such as Microsoft Intune. For more information on Download mode options.

How Microsoft uses Delivery Optimization

At Microsoft, to help ensure that ongoing deployments weren't affecting our network and taking away bandwidth for other services, Microsoft IT used a couple of different bandwidth management strategies. Delivery Optimization, peer-to-peer caching enabled through Group Policy, was piloted and then deployed to all managed devices using Group Policy. Based on recommendations from the Delivery Optimization team, we used the "group" configuration to limit sharing of content to only the devices that are members of the same Active Directory domain. The content is cached for 24 hours. More than 76 percent of content came from peer devices versus the Internet.

Using a proxy with Delivery Optimization

If a proxy is being used in your environment, see Using a proxy with Delivery Optimization to understand the proxy settings needed to properly using Delivery Optimization.

Delivery Optimization client-service communication explained

To gain a deeper understanding of the Delivery Optimization client-service communication workflow, see Delivery Optimization client-service communication explained

Set up Delivery Optimization for Windows

Learn more about the Delivery Optimization settings to ensure proper setup in your environment.

Delivery Optimization reference

For a complete list of Delivery Optimization settings, see Delivery Optimization reference.

New in Windows 10, version 20H2 and Windows 11

See What's new in Delivery Optimization

Last updated on 12/03/2025

Windows Compatibility Cookbook

Article11/17/2021

Windows 10, Windows 8, Windows 8.1, and Windows Server 2012 R2 introduce the latest operating system technology and software development platforms for use by app developers and enterprises worldwide. As part of further enhancing the security, reliability, performance, and user experience of Windows, Microsoft has introduced many new features, improved existing features, and removed others.

While the goal of Windows 10, Windows 8, Windows 8.1, and Windows Server 2012 R2 is to stay highly compatible with most of their respective apps written for previously released operating systems, some compatibility breaks are inevitable due to innovations, tightened security, and increased reliability. Overall, the compatibility of the new operating systems with existing apps is high.

This *Cookbook* is divided into three major sections, Windows 10, Windows 8.1 and Windows 8:

- Windows 10
- Windows 8.1 and Windows Server 2012 R2
- Windows 8 and Windows Server 2012

The following topics are either new with this release of the *Cookbook* or have been updated:

- Windows Image File Boot (WimBoot)
- Windows App Certification Kit

We invite you to check out these topics to learn how to optimize your apps and take advantage of what this newest release of Windows has to offer.

Windows Compatibility Cookbook editions that address earlier versions of the Windows and Windows Server operating systems can be found at Windows Vista and Windows Server 2008 and Windows 7 and Windows Server 2008 R2 Application Quality Cookbook

Prepare for Windows 11

Article • 07/17/2024 • Applies to: ✅ Windows 11, ✅ Windows 10

Windows 10 and Windows 11 are designed to coexist so that you can use the same familiar tools and processes to manage both operating systems. Using a single management infrastructure that supports common applications across both Windows 10 and Windows 11 helps to simplify the migration process. You can analyze endpoints, determine application compatibility, and manage Windows 11 deployments in the same way that you do with Windows 10.

After you evaluate your hardware to see if it meets requirements for Windows 11, it's a good time to review your deployment infrastructure, tools, and overall endpoint and update management processes. Use this review time to look for opportunities to simplify and optimize. This article provides some helpful guidance to accomplish these tasks.

Infrastructure and tools

The tools that you use for core workloads during Windows 10 deployments can still be used for Windows 11.

> **Important**
>
> Be sure to check with the providers of any non-Microsoft solutions that you use. Verify compatibility of these tools with Windows 11, particularly if they provide security or data loss prevention capabilities.

On-premises solutions

- If you use Windows Server Update Services (WSUS), you need to sync the Windows 11 product category. After you sync the product category, you'll see Windows 11 offered as an option. If you want to validate Windows 11 builds before their broad release, you can also sync the **Windows Insider Pre-release** category.

  > **Note**

During deployment, you'll be prompted to agree to the Microsoft Software License Terms on behalf of your users. Additionally, you won't see an x86 option because Windows 11 isn't supported on 32-bit architecture.

- If you use Microsoft Configuration Manager, you can sync the **Windows 11** product category and begin upgrading eligible devices. If you want to validate Windows 11 builds before their broad release, you can also sync the **Windows Insider Pre-release** category.

> **Note**
>
> Configuration Manager will prompt you to accept the Microsoft Software License Terms on behalf of the users in your organization.

Cloud-based solutions

- If you use Windows Update client policies policies, you need to use the **Target Version** capability. This option is either through policy or Windows Autopatch. You need to use this option instead of only using feature update deferrals to upgrade from Windows 10 to Windows 11. Feature update deferrals are great for moving to newer versions of your current product. For example, Windows 10, version 21H2 to version 22H2. They don't automatically move devices between products, for example Windows 10 to Windows 11.
  - If you use Microsoft Intune and have a Microsoft 365 E3 license, use the feature update deployments page to select the latest version of Windows 11 and upgrade Windows 10 devices to Windows 11. You can also continue using the same update experience controls to manage Windows 10 and Windows 11 on the **Update Rings** page in Intune. If you aren't ready to move to Windows 11, keep the feature update version set at the version you're currently on. When you're ready to start upgrading devices, change the feature update deployment setting to specify Windows 11.
  - In group policy, **Select target Feature Update version** has two entry fields: **Product Version** and **Target Version**.
  - The product field must specify Windows 11 in order for devices to upgrade to Windows 11. If only the target version field is configured, the service offers the device matching versions of the same product.

- Quality update deferrals continue to work the same across both Windows 10 and Windows 11. This behavior is true regardless of which management tool you use to configure Windows Update client policies.

> **Note**
>
> Endpoints managed by Windows Update client policies don't automatically upgrade to Windows 11 unless an administrator explicitly configures a **Target Version** using the TargetReleaseVersion setting using a Windows CSP, a feature update profile in Intune, or the Select target Feature Update version setting group policy.

Cloud-based management

The cloud-based management capabilities of the Microsoft Intune family of products help consolidate device management and endpoint security into a single platform. Microsoft Intune also supports the diverse bring-your-own-device (BYOD) ecosystem that's common with hybrid work scenarios. It can also enable you to track your progress against compliance and business objectives while protecting user data.

The following are some common use cases and the corresponding Microsoft Intune capabilities that support them:

- **Provision and pre-configure new Windows 11 devices**: Windows Autopilot enables you to deploy new Windows 11 devices in a business-ready state that includes your desired applications, settings, and policies. It can also be used to change the edition of Windows. For example, you can upgrade from Professional to Enterprise edition and gain the use of advanced features.

- **Configure rules and control settings for users, apps, and devices**: When you enroll devices in Microsoft Intune, you have full control over apps, settings, features, and security for both Windows 11 and Windows 10. You can also use app protection policies to require multifactor authentication (MFA) for specific apps.

- **Streamline device management for frontline, remote, and onsite workers**: Cloud configuration is a standard, easy-to-manage, device configuration that is cloud-optimized for users with specific workflow needs. You can use Microsoft Intune to deploy it to devices running the Pro, Enterprise, and Education editions of Windows 11.

If you're exclusively using an on-premises device management solution like Configuration Manager, you can still use the cloud management gateway, enable tenant attach, or enable co-management with Microsoft Intune. These solutions can make it easier to keep devices secure and up-to-date.

Review servicing approach and policies

Every organization transitions to Windows 11 at its own pace. Microsoft is committed to supporting you through your migration to Windows 11, whether you're a fast adopter or will make the transition over the coming months or years.

When you think of OS updates as an ongoing process, you improve your ability to deploy updates. This approach enables you to stay current with less effort, and less effect on productivity. To begin, think about how you roll out Windows feature updates today: which devices, and at what pace.

Next, craft a deployment plan for Windows 11 that includes deployment groups, rings, users, or devices. There are no absolute rules for exactly how many rings to have for your deployments, but the following example is a common structure:

- Preview (first or canary): Planning and development
- Limited (fast or early adopters): Pilot and validation
- Broad (users or critical): Wide deployment

For more information, see Create a deployment plan.

Review policies

Review deployment-related policies, and take into consideration your organization's security objectives, update compliance deadlines, and device activity. Apply changes where you can gain a clear improvement, particularly regarding the speed of the update process or security.

Validate apps and infrastructure

To validate that your apps, infrastructure, and deployment processes are ready for Windows 11, join the Windows Insider Program for Business. Then opt into the Release Preview Channel.

If you use Windows Server Update Services (WSUS), you can deploy directly from the Windows Insider Prerelease category using one of the following processes:

- Set **Manage Preview Builds** to **Release Preview** in Windows Update client policies.
- Use Azure Virtual Desktop and Azure Marketplace images.
- Download and deploy ISOs from Microsoft's Windows Insider Program ISO download page.

Regardless of the method you choose, you have the benefit of free Microsoft support when validating prerelease builds. Free support is available to any commercial customer deploying Windows 10 or Windows 11 Preview Builds, once they become available through the Windows Insider Program.

Analytics and assessment tools

If you use Microsoft Intune and have onboarded devices to Endpoint analytics, you have access to a hardware readiness assessment. This tool enables you to quickly identify which of your managed devices are eligible for the Windows 11 upgrade.

Prepare a pilot deployment

A pilot deployment is a proof of concept that rolls out an upgrade to a select number of devices in production, before deploying it broadly across the organization.

At a high level, the tasks involved are:

1. Assign a group of users or devices to receive the upgrade.
2. Implement baseline updates.
3. Implement operational updates.
4. Validate the deployment process.
5. Deploy the upgrade to devices.
6. Test and support the pilot devices.
7. Determine broad deployment readiness based on the results of the pilot.

User readiness

Don't overlook the importance of user readiness to deliver an effective, enterprise-wide deployment of Windows 11. Windows 11 has a familiar design, but your users will see several enhancements to the overall user interface. They'll also need to adapt to changes in menus and settings pages. Therefore, consider the following tasks to prepare users and IT support staff for Windows 11:

- Create a communications schedule to ensure that you provide the right message at the right time to the right groups of users, based on when they'll see the

changes.

- Draft concise emails that inform users of what changes they can expect to see. Offer tips on how to use or customize their experience. Include information about support and help desk options.
- Update help desk manuals with screenshots of the new user interface, the out-of-box experience for new devices, and the upgrade experience for existing devices.

For more information and resources, see the Meet Windows 11 video series.

See also

Stay current with Windows devices and Microsoft 365 Apps

Plan for Windows 11

Windows help & learning for users

Feedback

Was this page helpful? Yes No

Provide product feedback

Prepare to deploy Windows

Article • 02/23/2023 • Applies to: ✅ Windows 11, ✅ Windows 10

Having worked through the activities in the planning phase, you should be in a good position to prepare your environment and process to deploy Windows client. The planning phase left you with these useful items:

- A clear understanding of necessary personnel and their roles and criteria for rating app readiness
- A plan for testing and validating apps
- An assessment of your deployment infrastructure and definitions for operational readiness
- A deployment plan that defines the rings you want to use

Now you're ready to actually start making changes in your environment to get ready to deploy.

Prepare infrastructure and environment

- Deploy site server updates for Configuration Manager.
- Update non-Microsoft security tools like security agents or servers.
- Update non-Microsoft management tools like data loss prevention agents.

Your infrastructure probably includes many different components and tools. You need to ensure your environment isn't affected by issues due to the changes you make to the various parts of the infrastructure. Follow these steps:

1. Review all of the infrastructure changes that you've identified in your plan. It's important to understand the changes that need to be made and to detail how to implement them. This process prevents problems later on.

2. Validate your changes. You validate the changes for your infrastructure's components and tools, to help you understand how your changes could affect your production environment.

3. Implement the changes. Once the changes have been validated, you can implement the changes across the wider infrastructure.

You should also look at your organization's environment's configuration and outline how you'll implement any necessary changes previously identified in the plan phase to support the update. Consider what you need to do for the various settings and policies that currently underpin the environment. For example:

- Implement new draft security guidance. New versions of Windows can include new features that improve your environment's security. Your security teams will want to make appropriate changes to security-related configurations.
- Update security baselines. Security teams understand the relevant security baselines and will have to work to make sure all baselines fit into whatever guidance they have to adhere to.

However, your configuration will consist of many different settings and policies. It's important to only apply changes where they're necessary, and where you gain a clear improvement. Otherwise, your environment might face issues that slow down the update process. You want to ensure your environment isn't affected adversely because of changes you make. For example:

1. Review new security settings. Your security team reviews the new security settings to understand how they can best be set to facilitate the update, and to also investigate the potential effects they might have on your environment.
2. Review security baselines for changes. Security teams also review all the necessary security baselines, to ensure the changes can be implemented, and ensure your environment remains compliant.
3. Implement and validate security settings and baseline changes. Your security teams will then implement all of the security settings and baselines, having addressed any potential outstanding issues.

Prepare applications and devices

You've previously decided on which validation methods you want to use to validate apps in the upcoming pilot deployment phase. Now is a good time to make sure that individual devices are ready and able to install the next update without difficulty.

Ensure updates are available

Enable update services on devices. Ensure that every device is running all the services Windows Update relies on. Sometimes users or even malware can disable the services Windows Update requires to work correctly. Make sure the following services are running:

- Background Intelligent Transfer Service
- Background Tasks Infrastructure Service
- BranchCache (if you use this feature for update deployment)

- ConfigMgr Task Sequence Agent (if you use Configuration Manager to deploy updates)
- Cryptographic Services
- DCOM Server Process Launcher
- Device Install
- Delivery Optimization
- Device Setup Manager
- License Manager
- Microsoft Account Sign-in Assistant
- Microsoft Software Shadow Copy Provider
- Remote Procedure Call (RPC)
- Remote Procedure Call (RPC) Locator
- RPC Endpoint Mapper
- Service Control Manager
- Task Scheduler
- Token Broker
- Update Orchestrator Service
- Volume Shadow Copy Service
- Windows Automatic Update Service
- Windows Backup
- Windows Defender Firewall
- Windows Management Instrumentation
- Windows Management Service
- Windows Module Installer
- Windows Push Notification
- Windows Security Service
- Windows Time Service
- Windows Update
- Windows Update Medic Service

You can check these services manually by using Services.msc, or by using PowerShell scripts, or other methods.

Network configuration

Ensure that devices can reach necessary Windows Update endpoints through the firewall. For example, for Windows 10, version 2004, the following protocols must be able to reach these respective endpoints:

Expand table

Protocol	Endpoint URL
TLS 1.2	`*.prod.do.dsp.mp.microsoft.com`
HTTP	`*.dl.delivery.mp.microsoft.com`
HTTP	`*.windowsupdate.com`
HTTPS	`*.delivery.mp.microsoft.com`
TLS 1.2	`*.update.microsoft.com`
TLS 1.2	`tsfe.trafficshaping.dsp.mp.microsoft.com`

> **Note**
>
> Be sure not to use HTTPS for those endpoints that specify HTTP, and vice versa. The connection will fail.

The specific endpoints can vary between Windows versions. See, for example, Windows 10 2004 Enterprise connection endpoints. Similar articles for other Windows client versions are available in the table of contents nearby.

Optimize download bandwidth

Set up Delivery Optimization for peer network sharing or Microsoft Connected Cache.

Address unhealthy devices

In the course of surveying your device population, you might find devices that have systemic problems that could interfere with update installation. Now is the time to fix those problems.

- **Low disk space:** Quality updates require a minimum of 2 GB to successfully install. Feature updates require between 8 GB and 15 GB depending upon the configuration. On Windows 10, version 1903 and later (and Windows 11) you can proactively use the "reserved storage" feature (for wipe and loads, rebuilds, and new builds) to avoid running out of disk space. If you find a group of devices that don't have enough disk space, you can often resolve the problem by cleaning up log files and asking users to clean up data if necessary. A good place to start is to delete the following files:
  - C:\Windows\temp

- C:\Windows\cbstemp (though this file might be necessary to investigate update failures)
- C:\Windows\WindowsUpdate.log (though this file might be necessary to investigate update failures)
- C:\Windows.Old (these files should automatically clean up after 10 days or might ask the device user for permission to clean up sooner when constrained for disk space)

You can also create and run scripts to perform additional cleanup actions on devices, with administrative rights, or use Group Policy settings.

- Clean up the Windows Store Cache by running C:\Windows\sytem32\wsreset.exe.
- Optimize the WinSxS folder on the client machine by using **Dism.exe /online /Cleanup-Image /StartComponentCleanup**.
- Compact the operating system by running **Compact.exe /CompactOS:always**.
- Remove Windows Features on Demand that the user doesn't need. For more information, see Features on Demand.
- Move Windows Known Folders to OneDrive. For more information, see Use Group Policy to control OneDrive sync settings.
- Clean up the Software Distribution folder. Try deploying these commands as a batch file to run on devices to reset the download state of Windows Updates:

  Console

  ```
  net stop wuauserv
  net stop cryptSvc
  net stop bits
  net stop msiserver
  ren C:\Windows\SoftwareDistribution C:\Windows\SoftwareDistribution.old
  net start wuauserv
  net start cryptSvc
  net start bits
  net start msiserver
  ```

- **Application and driver updates:** Out-of-date app or driver software can prevent devices from updating successfully. Deploy any updates from the vendor(s) for any problematic application or driver versions to resolve issues.
- **Corruption:** In rare circumstances, a device that has repeated installation errors might be corrupted in a way that prevents the system from applying a new update.

You might have to repair the Component-Based Store from another source. You can fix the problem with the System File Checker.

Prepare capability

In the plan phase, you determined the specific infrastructure and configuration changes that needed to be implemented to add new capabilities to the environment. Now you can move on to implementing those changes defined in the plan phase. You need to complete these higher-level tasks to gain those new capabilities:

- Enable capabilities across the environment by implementing the changes. For example, implement updates to relevant ADMX templates in Active Directory. New Windows versions come with new policies that you use to update ADMX templates.
- Validate new changes to understand how they affect the wider environment.
- Remediate any potential problems that have been identified through validation.

Prepare users

Users often feel like they're forced into updating their devices randomly. They often don't fully understand why an update is needed, and they don't know when updates would be applied to their devices ahead of time. It's best to ensure that upcoming updates are communicated clearly and with adequate warning.

You can employ various measures to achieve this goal, for example:

- Send overview email about the update and how it will be deployed to the entire organization.
- Send personalized emails to users about the update with specific details.
- Set an opt-out deadline for employees that need to remain on the current version for a bit longer, due to a business need.
- Provide the ability to voluntarily update at users' convenience.
- Inform users of a mandatory installation date when the update will be installed on all devices.

Feedback

Was this page helpful? Yes No

Provide product feedback

Policies for update compliance, activity, and user experience

Applies to: ✅ Windows 11, ✅ Windows 10

Keeping devices up to date is the best way to keep them working smoothly and securely.

Deadlines for update compliance

You can control how strictly devices must reliably keep to your desired update schedule by using update deadline policies. Windows components adapt based on these deadlines. Also, they can make tradeoffs between user experience and velocity in order to meet your desired update deadlines. For example, they can prioritize user experience well before the deadline approaches, and then prioritize velocity as the deadline nears, while still affording the user some control.

Deadlines

Beginning with Windows 10, version 1903 and with the August 2019 security update for Windows 10, version 1709 and later (including Windows 11), a new policy was introduced to replace older deadline-like policies: **Specify deadlines for automatic updates and restarts**.

The older policies started enforcing deadlines once the device reached a `restart pending` state for an update. The new policy starts the countdown for the update installation deadline from when the update is published plus any deferral. In addition, this policy includes a configurable grace period and the option to opt out of automatic restarts until the deadline is reached (although we recommend always allowing automatic restarts for maximum update velocity).

We recommend you set deadlines as follows:

- Quality update deadline, in days: 2
- Feature update deadline, in days: 2

Notifications are automatically presented to the user at appropriate times, and users can choose to be reminded later, to reschedule, or to restart immediately, depending on how close the deadline is. We recommend that you do **not** set any notification policies, because they're automatically configured with appropriate defaults. An exception is if you have kiosks or digital signage.

While three days for quality updates and seven days for feature updates is our recommendation, you might decide you want more or less, depending on your organization and its requirements, and this policy is configurable down to a minimum of two days.

> **Important**
>
> If the device is unable to reach the Internet, it can't determine when Microsoft published the update, so it won't be able to enforce the deadline. Learn more about low activity devices.

Grace periods

You can set a period of days for Windows to find a minimally disruptive automatic restart time before the restart is enforced. This is especially useful in cases where a user has been away for many days (for example, on vacation) so that the device won't be forced to update immediately when the user returns.

We recommend you set the following:

- Grace period, in days: 5

Once the deadline and grace period have passed, updates are applied automatically, and a restart occurs regardless of active hours.

Let Windows choose when to restart

Windows can use user interactions to dynamically identify the least disruptive time for an automatic restart. To take advantage of this feature, ensure **ConfigureDeadlineNoAutoReboot** is set to **Disabled**.

Device activity policies

Windows typically requires that a device is active and connected to the internet for at least six hours, with at least two of continuous activity, in order to successfully complete a system update. The device could have other physical circumstances that prevent successful installation of an update--for example, if a laptop is running low on battery power, or the user has shut down the device before active hours end and the device can't comply with the deadline.

You can use the settings in this section to ensure that devices are available to install updates during the update compliance period.

Active hours

"Active hours" identify the period of time when a device is expected to be in use. Normally, restarts occur outside of these hours. Windows 10, version 1903 introduced "intelligent active hours," which allow the system to learn active hours based on a user's activities, rather than you as an administrator having to make decisions for your organization or allowing the user to choose active hours that minimize the period when the system can install an update.

> **Important**
>
> If you used the **Configure Active Hours** setting in previous versions of Windows 10, these options must be **Disabled** in order to take advantage of intelligent active hours.

If you do set active hours, we recommend setting the following policies to **Disabled** in order to increase update velocity:

- Delay automatic reboot. While it's possible to set the system to delay restarts for users who are logged in, this setting might delay an update indefinitely if a user is always either logged in or shut down. Instead, we recommend setting the following policies to **Disabled**:
  - **Turn off auto-restart during active hours**
  - **No auto-restart with logged on users for scheduled automatic updates**

  > **Note**
  >
  > The **No auto-restart with logged on users for scheduled automatic updates installation** policy was never created as a CSP. In Group Policy this policy doesn't work exactly as per description. This policy can result in no quality update reboots period, given many users never log off. The recommendation to replace this would be to leverage compliance deadline and then to configure no-auto reboot to prevent non-user aware reboots prior to the deadline being reached. For server devices, leverage Configure Automatic Updates options 7 - notify to install and notify to reboot.

- Limit restart delays. By using compliance deadlines, your users receive notifications that updates will occur, so we recommend that you set this policy to **Disabled**, to allow compliance deadlines to eliminate the user's ability to delay a restart outside of compliance deadline settings.

- **Do not allow users to approve updates and reboots**. Letting users approve or engage with the update process outside of the deadline policies decreases update velocity and increases risk. These policies should be set to **Disabled**:
  - Update/RequireUpdateApproval

    - This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.
  - Update/EngagedRestartDeadline
    - This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.
  - Update/EngagedRestartDeadlineForFeatureUpdates
    - This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.
  - Update/EngagedRestartSnoozeSchedule
    - This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.
  - Update/EngagedRestartSnoozeScheduleForFeatureUpdates
    - This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.
  - Update/EngagedRestartTransitionSchedule
    - This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

- Configure automatic update. By properly setting policies to configure automatic updates, you can increase update velocity by having clients contact a Windows Server Update Services (WSUS) server so it can manage them. We recommend that you set this policy to **Disabled**. However, if you need to provide values, ensure that you set downloads to install automatically by setting the Group Policy to **4**. If you're using Microsoft Intune, setting the value to Reset to Default.

- **Allow auto Windows Update to download over metered networks**. Since more devices primarily use cellular data and don't have wi-fi access, consider allowing users to automatically download updates from a metered network. Though the default setting doesn't allow download over a metered network, setting this value to **1** can increase velocity by enabling users to get updates whether they're connected to the internet or not, provided they have cellular service.

> **Important**
>
> Older versions of Windows don't support intelligent active hours. If your device runs a version of Windows prior to Windows 10, version 1903, we recommend setting the following policies:
>
> - Configure active hours. Starting with Windows 10, version 1703, you can specify a maximum active-hour range which is counted from the active hours start time. We recommend setting this value to **10**.

- Schedule update installation. In the **Configure Automatic Updates** settings, there are two ways to control a forced restart after a specified installation time. If you use **schedule update installation**, do not enable both settings because they will most likely conflict.
  - **Specify automatic maintenance time**. This setting lets you set broader maintenance windows for updates and ensures that this schedule does not conflict with active hours. We recommend setting this value to **3** (corresponding to 3 AM). If 3:00 AM is in the middle of the work shift, pick another time that is at least a couple hours before your scheduled work time begins.
  - **Schedule the install time**. This setting allows you to schedule an installation time for a restart. We do *not* recommend you set this to **Disabled** as it could conflict with active hours.

Power policies

Devices must actually be available during nonactive hours in order to an update. They can't do this if power policies prevent them from waking up. In our organization, we strive to set a balance between security and eco-friendly configurations. We recommend the following settings to achieve what we feel are the appropriate tradeoffs:

To a user, a device is either on or off, but for Windows, there are states that allow an update to occur (active) and states that don't (inactive). Some states are considered active (sleep), but the user may think the device is off. Also, there are power statuses (plugged in/battery) that Windows checks before starting an update.

You can override the default settings and prevent users from changing them in order to ensure that devices are available for updates during nonactive hours.

> **Note**
>
> One way to ensure that devices can install updates when you need them to is to educate your users to keep devices plugged in during non-active hours. Even with the best policies, a device that isn't plugged in will not be updated, even in sleep mode.

We recommend these power management settings:

- Sleep mode (S1 or S0 Low Power Idle or Modern Standby). When a device is in sleep mode, the system appears to be off but if an update is available, it can wake up the device in order to take an update. The power consumption in sleep mode is between

working (system fully usable) and hibernate (S4 - lowest power level before shutdown). When a device isn't being used, the system will generally move to sleep mode before it goes to hibernate. Issues in velocity arise when the time between sleep and hibernate is too short and Windows doesn't have time to complete an update. Sleep mode is an important setting because the system can wake the system from sleep in order to start the update process, as long as there's enough power.

Set the following policies to **Enable** or **Do Not Configure** in order to allow the device to use sleep mode:

- Power/AllowStandbyStatesWhenSleepingOnBattery
- Power/AllowStandbyWhenSleepingPluggedIn

Set the following policies to **1 (Sleep)** so that when a user closes the lid of a device, the system goes to sleep mode and the device has an opportunity to take an update:

- Power/SelectLidCloseActionOnBattery
- Power/SelectLidCloseActionPluggedIn
- **Hibernate**. When a device is hibernating, power consumption is low and the system can't wake up without user intervention, like pressing the power button. If a device is in this state, it can't be updated unless it supports an ACPI Time and Alarm Device (TAD). That said, if a device supporting Traditional Sleep (S3) is plugged in, and a Windows update is available, a hibernate state is delayed until the update is complete.

> **Note**
>
> This does not apply to devices that support Modern Standby (S0 Low Power Idle). You can check which system sleep state (S3 or S0 Low Power Idle) a device supports by running `powercfg /a` at a command prompt. For more, see Powercfg options.

The default timeout on devices that support traditional sleep is set to three hours. We recommend that you don't reduce these policies in order to allow Windows Update the opportunity to restart the device before sending it into hibernation:

- Power/HibernateTimeoutOnBattery
- Power/HibernateTimeoutPluggedIn

Old or conflicting policies

Each release of Windows client can introduce new policies to make the experience better for both administrators and their organizations. When we release a new client policy, we either

release it purely for that release and later or we backport the policy to make it available on earlier versions.

> **Important**
>
> If you are using Group Policy, note that we don't update the old ADMX templates and you must use the newer (1903) ADMX template in order to use the newer policy. Also, if you are using an MDM tool (Microsoft or non-Microsoft), you can't use the new policy until it's available in the tool interface.

As administrators, you have set up and expect certain behaviors, so we expressly don't remove older policies since they were set up for your particular use cases. However, if you set a new policy without disabling a similar older policy, you could have conflicting behavior and updates might not perform as expected.

> **Important**
>
> We sometimes find that administrators set devices to get both Group Policy settings and MDM settings from an MDM server such as Microsoft Intune. Policy conflicts are handled differently, depending on how they are ultimately set up:
>
> - Windows updates: Group Policy settings take precedence over MDM.
> - Microsoft Intune: If you set different values for the same policy on two different groups, you will receive an alert and neither policy will be set until the conflict is resolved. It is crucial that you disable conflicting policies in order for devices in your organization to take updates as expected. For example, if a device is not reacting to your MDM policy changes, check to see if a similar policy is set in Group Policy with a differing value. If you find that update velocity is not as high as you expect or if some devices are slower than others, it might be time to clear all policies and settings and specify only the recommended update policies. See the Policy and settings reference for a consolidated list of recommended policies.

The following are policies that you might want to disable because they could decrease update velocity or there are better policies to use that might conflict:

- **Defer Feature Updates Period in Days**. For maximum update velocity, it's best to set this to **0** (no deferral) so that the feature update can complete and monthly security updates are offered again. Even if there's an urgent quality update that must be quickly deployed, it's best to use **Pause Feature Updates** rather than setting a deferral policy. You can choose a longer period if you don't want to stay up to date with the latest feature update.

- **Defer Quality Updates Period in Days**. To minimize risk and maximize update velocity, the maximum time you might want to consider while evaluating the update with a different ring of devices is two to three days.
- **Pause Feature Updates Start Time**. Set to **Disabled** unless there is a known issue requiring time for a resolution.
- **Pause Quality Updates Start Time**. Set to **Disabled** unless there's a known issue requiring time for a resolution.
- **Deadline No Auto Reboot**. Default is **Disabled - Set to 0** . We recommend that devices automatically try to restart when an update is received. Windows uses user interactions to dynamically identify the least disruptive time to restart.

There are also additional policies that are no longer supported or have been superseded.

Last updated on 09/30/2025

Update Baseline

Article • 08/24/2023 • Applies to: ✅ Windows 10

> **Note**
>
> Update Baseline isn't currently available for Windows 11.

With the large number of different policies offered for Windows client, Update Baseline provides a clear list of recommended Windows Update policy settings for IT administrators who want the best user experience while also meeting their monthly update compliance goals. See Policies included in the Update Baseline for the full list of policy configurations.

Why is Update Baseline needed?

Update Baseline is an industry-tested solution that improves update adoption rates while also maintaining a high-quality user experience. Whether you're just starting out, or you have been configuring policies for years, Update Baseline can help get you to a known good state with an excellent user experience. Applying the baseline is especially helpful for organizations that have many years of policy configurations to clear out lingering misconfigurations.

You can use Update Baseline to:

- Ensure that user and device configuration settings are compliant with the baseline.
- Set configuration settings. You can use Group Policy to configure a device with the setting values specified in the baseline.

Update Baseline doesn't affect your offering policies, whether you're using deferrals or target version to manage which updates are offered to your devices and when.

Policies included in the Update Baseline

The Update Baseline configures settings in these Group Policy areas:

- System/Power Management
- Windows Components/Delivery Optimization
- Windows Components/Windows Update

For the complete detailed list of all settings and their values, see the MSFT Windows Update.htm file in the Update Baseline toolkit at the Download Center

How do I get started?

The Update Baseline toolkit makes it easy by providing a single command for IT Admins to load the baseline settings into Group Policy Management Console. You can get the Update Baseline toolkit (included as a part of the Security Compliance Toolkit) from the Download Center.

Today, the Update Baseline toolkit is currently only available for use with Group Policy.

Feedback

Was this page helpful? Yes No

Provide product feedback

Configure Delivery Optimization (DO) for Windows

Applies to: ✅ Windows 11, ✅ Windows 10, ✅ Delivery Optimization

This article describes the different configuration considerations to optimize Delivery Optimization (DO) in your environment.

Delivery Optimization set up considerations

Use this checklist to guide you through different aspects when modifying Delivery Optimization configurations for your environment.

1. Prerequisites to allow Delivery Optimization communication
2. Evaluate Delivery Optimization policies based on the following items:
    - Network topology
    - Organization size
    - System resources
    - Improve P2P efficiencies
3. Using Microsoft Connected Cache
4. Choose where to set Delivery Optimization policies

1. Prerequisites to allow Delivery Optimization communication

Delivery Optimization (DO) is used to download Microsoft content from different sources (HTTP source, peers, and/or dedicated cache solution). It requires communication between the

DO client and services to find the best and most reliable sources of content. For this technology to work, the DO client running on the Windows device must be able to reach the DO cloud service.

Find out more about the requirements for Firewall, Proxy, and Port settings to enable Delivery Optimization communication.

Firewall

There are service endpoints that you need to permit through your Firewall to communicate with the Delivery Optimization service. Check the full list of endpoints to permit.

Proxy

To allow peer-to-peer (P2P) to work properly, you need to allow direct calls to the Delivery Optimization service from your devices. When using a proxy, you want to bypass calls from the Delivery Optimization service (*.prod.do.dsp.mp.microsoft.com).

Local proxy

For downloads from HTTP sources, Delivery Optimization can use the automatic proxy discovery capability of WinHttp to handle communication with the proxy server. It's important to know that Delivery Optimization uses byte range requests, so you'll want to make sure your proxy allows this capability. Learn more about using Delivery Optimization with a proxy server.

Cloud proxy

If you're using a cloud proxy, the calls to the DO service are funneled through your cloud proxy and the public IP address of the devices is altered, preventing P2P from working properly. To avoid any issues, you should configure it to allow Delivery Optimization traffic to bypass the proxy. Otherwise, you may experience reduced performance and increased bandwidth consumption.

Ports

Delivery Optimization requires the use of certain ports to deliver content. Make sure all the required ports are open to make Delivery Optimization work seamlessly.

Expand table

Port	Protocol	Function
7680	TCP/IP	Listen for P2P using TCP/IP
3544	UDP	Use Teredo to discover and connect to peers across NATs. For more information, see the Teredo documentation.
443	HTTPS / TLS 1.2	Use to communicate Delivery Optimization client and service

2. Evaluate Delivery Optimization policies

There are a range of Delivery Optimization settings available to meet the needs of your environment. To fully leverage Delivery Optimization, you should focus on key areas to determine the most suitable settings for your unique environment.

2a. Network topology

Peer groups can be defined in Delivery Optimization using a combination of settings such as DODownloadMode, DOGroupID, DOGroupIDSource, and DORestrictPeerSelection. *The combination of settings used depends on your desired peer group(s) and your network topology.*

Peering setup options

Delivery Optimization can use P2P to help improve bandwidth efficiencies. The section outlines the different options available to define peer groups for your environment.

Local area network (LAN)

To define a peer group limited to your LAN, choose DODownloadMode (1), LAN-mode. This download mode setting includes any devices that share the same public IP address when they connect to the Internet (behind the same NAT) in a single peer group.

> **Note**
>
> Consider using Group download mode and/or limiting peer selection to the subnet if your network topology is a Hub and Spoke.

Wide area network (WAN)

To achieve peer groups across NATs within the same site, over the WAN, or to have more control in your local environment, use download mode '2', Group-mode. Group download mode allows you to define a unique GUID Group ID or use existing logical groupings (for example, AD Site) in your enterprise with the DOGroupIDSource setting to identify a peer group.

GroupIDSource default behavior

There are several options for identifying your Group ID using the DOGroupIDSource. The default behavior, when the DOGroupID or DOGroupIDSource policies aren't set, is to determine the Group ID using AD Site (1), Authenticated domain SID (2), or Microsoft Entra tenant ID (5). Learn more about all DOGroupIDSource available options.

> **Note**
>
> If your peer group spans across NATs, the Teredo service will be used on port 3544.
>
> For Configuration Manager users, we recommend leveraging existing **boundary groups** to define the peer groups.

Restrict peer selection

If your environment requires a more granular approach, you can use the restrict peer discovery setting alongside the download mode to achieve more control. For example, if you have several different subnets behind the same NAT but want to limit your peer groups to a single subnet, choose DODownloadMode (1) and DORestrictPeerSelection (Subnet). This setting can be used with any of the peer-related download modes (1, 2, or 3).

Nonpeering options

There are two valid download modes that don't use P2P functionality to deliver content; download modes (0) and (99). Download mode (0) uses additional metadata provided by the Delivery Optimization services for a peerless, reliable, and efficient download experience. Download mode (99) will provide a reliable download experience over HTTP from the download's original source or Microsoft, with no other checks.

Peering with VPN

By default, if Delivery Optimization detects a VPN, peering isn't used. To change this behavior, use the DOAllowVPNPeerCaching policy. The Delivery Optimization client looks in the network adapter's 'Description' and 'FriendlyName' strings to determine VPN usage. To allow greater flexibility for VPN identification, use the DOVpnKeywords to add descriptors for a particular VPN you use in your organization.

> **Note**
>
> The default keyword list is "VPN", "Secure", and "Virtual Private Network". For example, "MYVPN" matches the "VPN" keyword and would be detected as a VPN connection.

2b. Organization size

Delivery Optimization is designed to perform best in a large-scale environment with many devices. Depending on the size of the environment, you should evaluate the value of the DOMinFileSizeToCache to optimize peering.

Minimum file size to cache

Content peering has a limited number of slots available at any given time. By default, only content files that are 50 MB or larger can be used for peering. In an environment with more than 30 devices, change the DOMinFileSizeToCache to a lower value (10 MB), to have more

content available for peering, which can be distributed among many devices. For an even larger number of devices (>100), you can raise this setting to (1 MB).

Mobile devices

By default, P2P capabilities aren't enabled for devices using a battery. If there are many mobile devices in your environment, consider enabling the DOMinBatteryPercentageAllowedToUpload policy to 60%, to use peering while on battery.

2c. System resources

There are some Delivery Optimization configurations that can make an impact when various system resources are available.

Disk size

Alter the minimum disk size (default is 32 GB) a device must have to use peering with the DOMinDiskSizeAllowedToPeer setting.

Optimize cache size

You can also manage the amount of space the Delivery Optimization cache uses with the following settings: DOMaxCacheSize (default is 20%) and DOAbsoluteMaxCacheSize (default isn't configured).

RAM size

Control the minimum amount of RAM (inclusive) allowed to use peer caching (default is 4 GB), with DOMinRAMAllowedToPeer.

Large number of devices with idle system resources

In an environment with devices that are plugged in and have ample free disk space try increasing the content expiration interval of DOMaxCacheAge to seven or more (up to 30

days). You can take advantage of these devices, using them as excellent upload sources to upload more content over a longer period.

2d. Improve P2P efficiency

Looking to improve P2P efficiency? Some of the most powerful settings you can change that could have a significant impact within your environment include:

- Help optimize peer connection over HTTP connections using the DOMinBackgroundQoS policy. A good value for the DOMinBackgroundQoS policy is something lower than the average download speed seen in your network. For example, if your average speed is 1000 KB/s, set this policy to 500 KB/s.

- Improve chances of downloading from peers and/or cache server by delaying the time DO attempts to make connections before falling back to the HTTP source. The set of delay-related policies include:
  - DODelayBackgroundDownloadFromHttp
  - DODelayForegroundDownloadFromHttp

  To improve efficiencies from peers or a dedicated cache server, a good starting point is 60 seconds for background settings and 30 seconds for foreground settings.

> **Note**
>
> Not all content types are eligible for P2P. Refer to the complete list to learn more.

Bandwidth throttling options

Regardless of P2P, consider setting the following policies to avoid network disruption.

- Manage network usage as a percentage or absolute value. These policies include:
  - DOPercentageMaxBackgroundBandwidth
  - DOPercentageMaxForegroundBandwidth
  - DOMaxBackgroundDownloadBandwidth
  - DOMaxForegroundDownloadBandwidth
- Reduce disruptions by throttling differently at different times of day, using the following business hours policies:

- DOSetHoursToLimitBackgroundDownloadBandwidth
- DOSetHoursToLimitForegroundDownloadBandwidth.

> **Note**
>
> The absolute policies are recommended in low bandwidth environments.

3. Using Connected Cache

> **Note**
>
> When multiple Connected Cache servers are configured, the Delivery Optimization client connects to them in the order they appear in the list. If you've enabled delay settings for cache server fallback, the fallback-to-HTTP source delay applies to each new file download. This delay is not tied to any specific cache server.

- DOCacheHost is the list of cache host server names, separated with commas.
- DOCacheHostSource can be used to dynamically discover cache host servers on the network, using DHCP.
- DelayCacheServerFallbackBackground and DelayCacheServerFallbackForeground are the delay policies to help improve chances of pulling content from the network cache host servers. (See recommended values in Improve P2P efficiency section above).
- DODisallowCacheServerDownloadsOnVPN allows control of the cache host server to supply content, when device is on a VPN connection.

4. Choose where to set Delivery Optimization policies

Group Policies

Use Group Policy to manage Delivery Optimization settings here,

`Computer Configuration\Administrative Templates\Windows Components\Delivery Optimization`

MDM Policies

Use MDM to manage Delivery Optimization settings here,

`.Vendor/MSFT/Policy/Config/DeliveryOptimization/`

Delivery Optimization is integrated with both Microsoft Endpoint Manager and Configuration Manager.

- Microsoft Endpoint Manager (MEM)
- Microsoft Endpoint Configuration Manager (MECM)

Summary of basic configuration recommendations

Expand table

Use case	Policy	Recommended value
Use P2P	DownloadMode	1 or 2
Don't use P2P	DownloadMode	0
Number of devices in the organization	MinFileSizeToCache	1 MB for peer group > 100 devices
Idle system resources	MaxCacheAge	7 days (604800 seconds)
Improve P2P efficiency	MinBackgroundQoS and DelayBackgroundDownloadFromHttp / DelayForegroundDownloadFromHttp	500 KB/s and 60/30 seconds
Using Connected Cache	DelayCacheServerFallbackBackground / DelayCacheServerFallbackForeground	60/30 seconds

Enterprise using Intune

For a straightforward overview of configuring Delivery Optimization in Intune, check out this enterprise-focused guide with helpful recommendations.

Monitor Delivery Optimization

Whether you opt for the default Delivery Optimization configurations or tailor them to suit your environment, you'll want to track the outcomes to see how they improve your efficiency. The following options are available to monitor Delivery Optimization:

- On clients, review the activity monitor, which displays a breakdown of downloads by source, average speed, and upload stats for the current month
  - **Windows 11**: Settings > Windows Update > Advanced Options > Delivery Optimization > Activity Monitor
  - **Windows 10**: Settings > Update & Security > Delivery Optimization > Activity Monitor
- Windows Update for Business reports offers a Delivery Optimization report. For more information, see Monitor Delivery Optimization.

Troubleshoot Delivery Optimization

There could be many different reasons why Delivery Optimization isn't working in your environment. Learn more about the DO Troubleshooter and common problems and solutions to help improve the experience of using Delivery Optimization.

Test Delivery Optimization

Learn more about guidance on basic testing scenarios to see how Delivery Optimization works.

Last updated on 08/04/2025

Configure BranchCache for Windows client updates

Article • 01/08/2024 • Applies to: ✅ Windows 10

> **Looking for consumer information?** See Windows Update: FAQ

BranchCache is a bandwidth-optimization feature that has been available since the Windows Server 2008 R2 and Windows 7 operating systems. Each client has a cache and acts as an alternate source for content that devices on its own network request. Windows Server Update Services (WSUS) and Microsoft Configuration Manager can use BranchCache to optimize network bandwidth during update deployment, and it's easy to configure for either of them. BranchCache has two operating modes: Distributed Cache mode and Hosted Cache mode.

- Distributed Cache mode operates like the Delivery Optimization feature in Windows client: each client contains a cached version of the BranchCache-enabled files it requests and acts as a distributed cache for other clients requesting that same file.

  > **Tip**
  >
  > Distributed Cache mode is preferred to Hosted Cache mode for Windows clients updates to get the most benefit from peer-to-peer distribution.

- In Hosted Cache mode, designated servers at specific locations act as a cache for files requested by clients in its area. Then, rather than clients retrieving files from a latent source, the hosted cache server provides the content on its behalf.

For detailed information about how Distributed Cache mode and Hosted Cache mode work, see BranchCache Overview.

Configure clients for BranchCache

Whether you use BranchCache with Configuration Manager or WSUS, each client that uses BranchCache must be configured to do so. You typically make your configurations through Group Policy. For step-by-step instructions on how to use Group Policy to configure BranchCache for Windows clients, see Client Configuration in the BranchCache Early Adopter's Guide.

In Windows 10, version 1607, the Windows Update Agent uses Delivery Optimization by default, even when the updates are retrieved from WSUS. When using BranchCache with Windows client, set the Delivery Optimization **Download mode** to '100' (Bypass) to allow clients to use the Background Intelligent Transfer Service (BITS) protocol with BranchCache instead. For instructions on how to use BranchCache in Distributed Cache mode with WSUS, see the section WSUS and Configuration Manager with BranchCache in Distributed Cache mode.

> **Note**
>
> Bypass Download mode (100) is only available in Windows 10 (starting in version 1607) and deprecated in Windows 11. BranchCache isn't supported for content downloaded using Delivery Optimization in Windows 11.

Configure servers for BranchCache

You can use WSUS and Configuration Manager with BranchCache in Distributed Cache mode. BranchCache in Distributed Cache mode is easy to configure for both WSUS and Microsoft Configuration Manager.

For a step-by-step guide to configuring BranchCache on Windows Server devices, see the BranchCache Deployment Guide (Windows Server 2012) or BranchCache Deployment Guide (Windows Server 2016).

In addition to these steps, there's one requirement for WSUS to be able to use BranchCache in either operating mode: the WSUS server must be configured to download updates locally on the server to a shared folder. This way, you can select BranchCache publication for the share. For Configuration Manager, you can enable BranchCache on distribution points; no other server-side configuration is necessary for Distributed Cache mode.

> **Note**
>
> Configuration Manager only supports Distributed Cache mode.

Feedback

Was this page helpful? Yes No

Provide product feedback

Prepare for Zero Touch Installation of Windows 10 with Configuration Manager

Article • 10/27/2022

Applies to:

- Windows 10

This article walks you through the Zero Touch Installation (ZTI) process of Windows 10 OS deployment using Microsoft Configuration Manager integrated with Microsoft Deployment Toolkit (MDT).

Prerequisites

In this article, you'll use components of an existing Configuration Manager infrastructure to prepare for Windows 10 OSD. In addition to the base setup, the following configurations should be made in the Configuration Manager environment:

- Configuration Manager current branch + all security and critical updates are installed.

  > **Note**
  >
  > Procedures in this guide use Configuration Manager version 1910. For more information about the versions of Windows 10 supported by Configuration Manager, see Support for Windows 10.

- The Active Directory Schema has been extended and System Management container created.
- Active Directory Forest Discovery and Active Directory System Discovery are enabled.
- IP range boundaries and a boundary group for content and site assignment have been created.
- The Configuration Manager reporting services point role has been added and configured.
- A file system folder structure and Configuration Manager console folder structure for packages has been created. Steps to verify or create this folder structure are provided below.
- The Windows ADK version that is supported for the version of Configuration Manager that is installed, including the Windows PE add-on. USMT should be installed as part of

the Windows ADK install.

- MDT version 8456
- DaRT 10 (part of MDOP 2015) are installed.
- The CMTrace tool (cmtrace.exe) is installed on the distribution point.

> **Note**
>
> CMTrace is automatically installed with the current branch of Configuration Manager at `Program Files\Microsoft Configuration Manager\tools\cmtrace.exe`.

For the purposes of this guide, we'll use three server computers: DC01, CM01 and HV01.

- DC01 is a domain controller and DNS server for the contoso.com domain. DHCP services are also available and optionally installed on DC01 or another server.
- CM01 is a domain member server and Configuration Manager software distribution point. In this guide, CM01 is a standalone primary site server.
- HV01 is a Hyper-V host computer that is used to build a Windows 10 reference image. This computer doesn't need to be a domain member.

All servers are running Windows Server 2019. However, an earlier, supported version of Windows Server can also be used.

All server and client computers referenced in this guide are on the same subnet. This interrelation isn't required, but each server and client computer must be able to connect to each other to share files, and to resolve all DNS names and Active Directory information for the contoso.com domain. Internet connectivity is also required to download OS and application updates.

Domain credentials

The following generic credentials are used in this guide. You should replace these credentials as they appear in each procedure with your credentials.

- **Active Directory domain name**: `contoso.com`
- **Domain administrator username**: `administrator`
- **Domain administrator password**: `pass@word1`

Create the OU structure

> **Note**
>
> If you've already created the OU structure that was used in the OSD guide for MDT, the same structure is used here and you can skip this section.

On **DC01**:

To create the OU structure, you can use the Active Directory Users and Computers console (dsa.msc), or you can use Windows PowerShell. The procedure below uses Windows PowerShell.

To use Windows PowerShell, copy the following commands into a text file and save it as `C:\Setup\Scripts\ou.ps1`. Ensure that you're viewing file extensions and that you save the file with the `.ps1` extension.

PowerShell

```
$oulist = Import-csv -Path c:\oulist.txt
ForEach($entry in $oulist){
    $ouname = $entry.ouname
    $oupath = $entry.oupath
    New-ADOrganizationalUnit -Name $ouname -Path $oupath -WhatIf
    Write-Host -ForegroundColor Green "OU $ouname is created in the location
$oupath"
}
```

Next, copy the following list of OU names and paths into a text file and save it as **C:\Setup\Scripts\oulist.txt**

text

```
OUName,OUPath
Contoso,"DC=CONTOSO,DC=COM"
Accounts,"OU=Contoso,DC=CONTOSO,DC=COM"
Computers,"OU=Contoso,DC=CONTOSO,DC=COM"
Groups,"OU=Contoso,DC=CONTOSO,DC=COM"
Admins,"OU=Accounts,OU=Contoso,DC=CONTOSO,DC=COM"
Service Accounts,"OU=Accounts,OU=Contoso,DC=CONTOSO,DC=COM"
Users,"OU=Accounts,OU=Contoso,DC=CONTOSO,DC=COM"
Servers,"OU=Computers,OU=Contoso,DC=CONTOSO,DC=COM"
Workstations,"OU=Computers,OU=Contoso,DC=CONTOSO,DC=COM"
Security Groups,"OU=Groups,OU=Contoso,DC=CONTOSO,DC=COM"
```

Lastly, open an elevated Windows PowerShell prompt on DC01 and run the ou.ps1 script:

PowerShell

```
Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Force
Set-Location C:\Setup\Scripts
.\ou.ps1
```

Create the Configuration Manager service accounts

A role-based model is used to configure permissions for the service accounts needed for operating system deployment in Configuration Manager. Perform the following steps to create the Configuration Manager **join domain** and **network access** accounts:

On **DC01**:

1. In the Active Directory Users and Computers console, browse to **contoso.com** > **Contoso** > **Service Accounts**.

2. Select the Service Accounts OU and create the CM_JD account using the following settings:

   - Name: CM_JD
   - User sign-in name: CM_JD
   - Password: `pass@word1`
   - User must change password at next logon: Clear
   - User can't change password: Selected
   - Password never expires: Selected

3. Repeat the step, but for the CM_NAA account.

4. After creating the accounts, assign the following descriptions:

   - CM_JD: Configuration Manager Join Domain Account
   - CM_NAA: Configuration Manager Network Access Account

Configure Active Directory permissions

In order for the Configuration Manager Join Domain Account (CM_JD) to join machines into the contoso.com domain, you need to configure permissions in Active Directory. These steps assume you've downloaded the sample Set-OUPermissions.ps1 script and copied it to `C:\Setup\Scripts` on DC01.

On **DC01**:

1. Sign in as contoso\administrator and enter the following commands at an elevated Windows PowerShell prompt:

   PowerShell

   ```
   Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Force
   Set-Location C:\Setup\Scripts
   .\Set-OUPermissions.ps1 -Account CM_JD -TargetOU
   "OU=Workstations,OU=Computers,OU=Contoso"
   ```

2. The Set-OUPermissions.ps1 script allows the CM_JD user account permissions to manage computer accounts in the Contoso / Computers / Workstations OU. The following list is that of permissions being granted:

   - Scope: This object and all descendant objects
   - Create Computer objects
   - Delete Computer objects
   - Scope: Descendant Computer objects
   - Read All Properties
   - Write All Properties
   - Read Permissions
   - Modify Permissions
   - Change Password
   - Reset Password
   - Validated write to DNS host name
   - Validated write to service principal name

Review the Sources folder structure

On **CM01**:

To support the packages you create in this article, the following folder structure should be created on the Configuration Manager primary site server (CM01):

- D:\Sources
- D:\Sources\OSD
- D:\Sources\OSD\Boot
- D:\Sources\OSD\DriverPackages
- D:\Sources\OSD\DriverSources
- D:\Sources\OSD\MDT
- D:\Sources\OSD\OS
- D:\Sources\OSD\Settings

- D:\Sources\OSD\Branding
- D:\Sources\Software
- D:\Sources\Software\Adobe
- D:\Sources\Software\Microsoft
- D:\Logs

> **Note**
>
> In most production environments, the packages are stored on a Distributed File System (DFS) share or a "normal" server share, but in a lab environment you can store them on the site server.

You can run the following commands from an elevated Windows PowerShell prompt to create this folder structure:

PowerShell

```
New-Item -ItemType Directory -Path "D:\Sources"
New-Item -ItemType Directory -Path "D:\Sources\OSD"
New-Item -ItemType Directory -Path "D:\Sources\OSD\Boot"
New-Item -ItemType Directory -Path "D:\Sources\OSD\DriverPackages"
New-Item -ItemType Directory -Path "D:\Sources\OSD\DriverSources"
New-Item -ItemType Directory -Path "D:\Sources\OSD\OS"
New-Item -ItemType Directory -Path "D:\Sources\OSD\Settings"
New-Item -ItemType Directory -Path "D:\Sources\OSD\Branding"
New-Item -ItemType Directory -Path "D:\Sources\OSD\MDT"
New-Item -ItemType Directory -Path "D:\Sources\Software"
New-Item -ItemType Directory -Path "D:\Sources\Software\Adobe"
New-Item -ItemType Directory -Path "D:\Sources\Software\Microsoft"
New-SmbShare -Name Sources$ -Path D:\Sources -FullAccess "NT
AUTHORITY\INTERACTIVE", "BUILTIN\Administrators"
New-Item -ItemType Directory -Path "D:\Logs"
New-SmbShare -Name Logs$ -Path D:\Logs -ChangeAccess EVERYONE
```

Integrate Configuration Manager with MDT

To extend the Configuration Manager console with MDT wizards and templates, install MDT with the default settings and run the **Configure ConfigManager Integration** desktop app. In these steps, we assume you've already downloaded MDT and installed it with default settings.

On **CM01**:

1. Sign in as contoso\administrator.

2. Ensure the Configuration Manager Console is closed before continuing.

3. Select Start, type **Configure ConfigManager Integration**, and run the application with the following settings:

    - Site Server Name: CM01.contoso.com
    - Site code: PS1

MDT integration with Configuration Manager.

Configure the client settings

Most organizations want to display their name during deployment. In this section, you configure the default Configuration Manager client settings with the Contoso organization name.

On **CM01**:

1. Open the Configuration Manager Console, select the **Administration** workspace, then select **Client Settings**.

2. In the right pane, right-click **Default Client Settings** and then select **Properties**.

3. In the **Computer Agent** node, in the **Organization name displayed in Software Center** text box, enter in **Contoso** and select **OK**.

Configure the organization name in client settings.

The Contoso organization name displayed during deployment.

Configure the Network Access account

Configuration Manager uses the Network Access account during the Windows 10 deployment process to access content on the distribution points. In this section, you configure the Network Access account.

On **CM01**:

1. Using the Configuration Manager Console, in the **Administration** workspace, expand **Site Configuration** and select **Sites**.

2. Right-click **PS1 - Primary Site 1**, point to **Configure Site Components**, and then select **Software Distribution**.

3. On the **Network Access Account** tab, select **Specify the account that accesses network locations** and add the account **CONTOSO\CM_NAA** as the Network Access account (password: **pass@word1**). Use the new **Verify** option to verify that the account can connect to the `\\DC01\sysvol` network share.

Test the connection for the Network Access account.

Enable PXE on the CM01 distribution point

Configuration Manager has many options for starting a deployment, but starting via PXE is certainly the most flexible in a large environment. In this section, you enable PXE on the CM01 distribution point.

On **CM01**:

1. In the Configuration Manager Console, in the **Administration** workspace, select **Distribution Points**.

2. Right-click the **\\CM01.CONTOSO.COM distribution point** and select **Properties**.

3. On the **PXE** tab, use the following settings:

- Enable PXE support for clients
- Allow this distribution point to respond to incoming PXE requests
- Enable unknown computer
- Require a password when computers use PXE
- Password and Confirm password: pass@word1

Configure the CM01 distribution point for PXE.

> ⓘ **Note**
>
> If you select **Enable a PXE responder without Windows Deployment Service**, then WDS won't be installed, or if it's already installed it will be suspended, and the **ConfigMgr PXE Responder Service** (**SccmPxe**) will be used instead of WDS. The ConfigMgr PXE Responder doesn't support multicast. For more information, see Install and configure distribution points.

4. Using the CMTrace tool, review the `C:\Program Files\Microsoft Configuration Manager\Logs\distmgr.log` file. Look for the **ConfigurePXE** and **CcmInstallPXE** lines.

The `distmgr.log` displays a successful configuration of PXE on the distribution point.

5. Verify that you've seven files in each of the folders `D:\RemoteInstall\SMSBoot\x86` and `D:\RemoteInstall\SMSBoot\x64`.

The contents of the D:\RemoteInstall\SMSBoot\x64 folder after you enable PXE.

> **Note**
>
> These files are used by WDS. They aren't used by the ConfigMgr PXE Responder. This article doesn't use the ConfigMgr PXE Responder.

Next, see Create a custom Windows PE boot image with Configuration Manager.

Components of Configuration Manager operating system deployment

Operating system deployment with Configuration Manager is part of the normal software distribution infrastructure, but there are more components. For example, operating system deployment in Configuration Manager may use the State Migration Point role, which isn't used by normal application deployment in Configuration Manager. This section describes the Configuration Manager components involved with the deployment of an operating system, such as Windows 10.

- **State migration point (SMP).** The state migration point is used to store user state migration data during computer replace scenarios.
- **Distribution point (DP).** The distribution point is used to store all packages in Configuration Manager, including the operating system deployment-related packages.
- **Software update point (SUP).** The software update point, which is normally used to deploy updates to existing machines, also can be used to update an operating system as part of the deployment process. You also can use offline servicing to update the image directly on the Configuration Manager server.
- **Reporting services point.** The reporting services point can be used to monitor the operating system deployment process.
- **Boot images.** Boot images are the Windows Preinstallation Environment (Windows PE) images Configuration Manager uses to start the deployment.
- **Operating system images.** The operating system image package contains only one file, the custom .wim image. This image is typically the production deployment image.
- **Operating system installers.** The operating system installers were originally added to create reference images using Configuration Manager. Instead, we recommend that you use MDT Lite Touch to create your reference images. For more information on how to create a reference image, see Create a Windows 10 reference image.

- **Drivers.** Like MDT Lite Touch, Configuration Manager also provides a repository (catalog) of managed device drivers.
- **Task sequences.** The task sequences in Configuration Manager look and feel much like the sequences in MDT Lite Touch, and they're used for the same purpose. However, in Configuration Manager, the task sequence is delivered to the clients as a policy via the Management Point (MP). MDT provides more task sequence templates to Configuration Manager.

> **Note**
>
> The Windows Assessment and Deployment Kit (ADK) for Windows 10 is also required to support management and deployment of Windows 10.

Why integrate MDT with Configuration Manager

As noted above, MDT adds many enhancements to Configuration Manager. While these enhancements are called Zero Touch, that name doesn't reflect how deployment is conducted. The following sections provide a few samples of the 280 enhancements that MDT adds to Configuration Manager.

> **Note**
>
> MDT installation requires the following:
>
> - The Windows ADK for Windows 10 (installed in the previous procedure)
> - Windows PowerShell (version 5.1 is recommended; type **$host** to check)
> - Microsoft .NET Framework

MDT enables dynamic deployment

When MDT is integrated with Configuration Manager, the task sequence processes more instructions from the MDT rules. In its most simple form, these settings are stored in a text file, the `CustomSettings.ini` file, but you can store the settings in Microsoft SQL Server databases, or have Microsoft Visual Basic Scripting Edition (VBScripts) or web services provide the settings used.

The task sequence uses instructions that allow you to reduce the number of task sequences in Configuration Manager and instead store settings outside the task sequence. Here are a few examples:

- The following settings instruct the task sequence to install the HP Hotkeys package, but only if the hardware is an HP EliteBook 8570w. You don't have to add the package to the task sequence.

```ini
[Settings]
Priority=Model
[HP EliteBook 8570w]
Packages001=PS100010:Install HP Hotkeys
```

- The following settings instruct the task sequence to put laptops and desktops in different organizational units (OUs) during deployment, assign different computer names, and finally have the task sequence install the Cisco VPN client, but only if the machine is a laptop.

```ini
[Settings]
Priority= ByLaptopType, ByDesktopType
[ByLaptopType]
Subsection=Laptop-%IsLaptop%
[ByDesktopType]
Subsection=Desktop-%IsDesktop%
[Laptop-True]
Packages001=PS100012:Install Cisco VPN Client
OSDComputerName=LT-%SerialNumber%
MachineObjectOU=ou=laptops,ou=Contoso,dc=contoso,dc=com
[Desktop-True]
OSDComputerName=DT-%SerialNumber%
MachineObjectOU=ou=desktops,ou=Contoso,dc=contoso,dc=com
```

The Gather action in the task sequence is reading the rules.

MDT adds an operating system deployment simulation environment

When testing a deployment, it's important to be able to quickly test any changes you make to the deployment without needing to run through an entire deployment. MDT rules can be tested quickly, saving significant testing time in a deployment project. For more information, see Configure MDT settings.

The folder that contains the rules, a few scripts from MDT, and a custom script (Gather.ps1).

MDT adds real-time monitoring

With MDT integration, you can follow your deployments in real time, and if you've access to Microsoft Diagnostics and Recovery Toolkit (DaRT), you can even remote into Windows Preinstallation Environment (Windows PE) during deployment. The real-time monitoring data can be viewed from within the MDT Deployment Workbench, via a web browser, Windows PowerShell, the Event Viewer, or Microsoft Excel 2013. In fact, any script or app that can read an Open Data (OData) feed can read the information.

View the real-time monitoring data with PowerShell.

MDT adds an optional deployment wizard

For some deployment scenarios, you may need to prompt the user for information during deployment such as the computer name, the correct organizational unit (OU) for the computer, or which applications should be installed by the task sequence. With MDT integration, you can enable the User-Driven Installation (UDI) wizard to gather the required information, and customize the wizard using the UDI Wizard Designer.

The optional UDI wizard opens in the UDI Wizard Designer.

MDT Zero Touch simply extends Configuration Manager with many useful built-in operating system deployment components. By providing well-established, supported solutions, MDT reduces the complexity of deployment in Configuration Manager.

Why use MDT Lite Touch to create reference images

You can create reference images for Configuration Manager in Configuration Manager, but in general it is recommended to create them in MDT Lite Touch for the following reasons:

- You can use the same image for every type of operating system deployment - Microsoft Virtual Desktop Infrastructure (VDI), Microsoft System Center Virtual Machine Manager (VMM), MDT, Configuration Manager, Windows Deployment Services (WDS), and more.
- Configuration Manager performs deployment in the LocalSystem context, which means that you can't configure the Administrator account with all of the settings that you would

like to be included in the image. MDT runs in the context of the Local Administrator, which means you can configure the look and feel of the configuration and then use the CopyProfile functionality to copy these changes to the default user during deployment.

- The Configuration Manager task sequence suppresses user interface interaction.
- MDT Lite Touch supports a Suspend action that allows for reboots, which is useful when you need to perform a manual installation or check the reference image before it's automatically captured.
- MDT Lite Touch doesn't require any infrastructure and is easy to delegate.

Related articles

Create a custom Windows PE boot image with Configuration Manager
Add a Windows 10 operating system image using Configuration Manager
Create an application to deploy with Windows 10 using Configuration Manager
Add drivers to a Windows 10 deployment with Windows PE using Configuration Manager
Create a task sequence with Configuration Manager and MDT
Deploy Windows 10 using PXE and Configuration Manager
Refresh a Windows 7 SP1 client with Windows 10 using Configuration Manager
Replace a Windows 7 SP1 client with Windows 10 using Configuration Manager

How to check Windows release health

Applies to: ✅ Windows 11, ✅ Windows 10

The Windows release health page in the Microsoft 365 admin center enables you to view the latest information on known issues for Windows monthly and feature updates. A known issue is an issue that impacts Windows devices and that has been identified in a Windows monthly update or feature update. The Windows release health page is designed to inform you about known issues. You can use this information to troubleshoot issues your users may be experiencing. You can also determine when, and at what scale, to deploy an update in your organization.

If you're unable to sign in to the Microsoft 365 admin portal, check the Microsoft 365 service health status page to check for known issues preventing you from signing into your tenant.

To be informed about the latest updates and releases, follow @WindowsUpdate on Twitter.

Prerequisites

Ensure the following prerequisites are met to display the Windows release health page in the Microsoft 365 admin center:

- One of the following licenses:
  - Windows 10/11 Enterprise E3 or E5 (included in some Microsoft 365 F3, E3, or E5 plans)
  - Windows 10/11 Education A3 or A5 (included in some Microsoft 365 A3 or A5 plans)

    Review the Product names and service plan identifiers for licensing to ensure that your product license contains at least one of the following service plan identifiers:
    - `e7c91390-7625-45be-94e0-e16907e03118`
    - `21b439ba-a0ca-424f-a6cc-52f954a5b111`
    - `8e229017-d77b-43d5-9305-903395523b99`
    - `dd353b54-910c-4d73-be7d-2ddc6e7b7332`
    - `c1cf417c-12dc-4265-b10a-168a0d537b5b`
    - `6998bd81-1f47-4d5d-badd-fb7efd875480`
    - `8e229017-d77b-43d5-9305-903395523b99`
    - `e7c91390-7625-45be-94e0-e16907e03118`
    - `d8c14701-c3d1-47a6-a034-fec01ac59dd3`
    - `21b439ba-a0ca-424f-a6cc-52f954a5b111`
    - `e041597c-9c7f-4ed9-99b0-2663301576f7`
    - `91d57b36-ce81-48ee-9a19-c751fdb09b00`

- `7bf960f6-2cd9-443a-8046-5dbff9558365`
- `8e2288ad-7af9-4b0c-af82-77504a7ab17d`

- Sign into the Microsoft 365 admin center using an admin role.
  - Most roles containing the word `administrator` give you access to the Windows release health page such as Helpdesk Administrator and Service Support Administrator. For more information, see Assign admin roles in the Microsoft 365 admin center.

> **Note**
>
> Currently, Windows release health is available for Government Community Cloud (GCC) tenants, but isn't available for GCC High and DoD.

How to review Windows release health information

1. Go to the Microsoft 365 admin center and sign in with an admin account.

2. To view Windows release health in the Microsoft 365 Admin Center, go to **Health** > **Windows release health**.

3. On the **Windows release health** page, you have access to known issue information for all supported versions of the Windows operating system.

   The **All versions** tab (the default view) shows all Windows products with access to their posted known issues.

A known issue is an issue that has been identified in a Windows monthly update or feature update that impacts Windows devices. The **Active and recently resolved** column provides a link to the **Known issues** tab filtered to the version selected. Selecting the **Known issues** tab shows known issues that are active or resolved within the last 30 days.

The **History** tab shows the history of known issues that have been resolved for up to 6 months.

The known issue summary provides the following information:

- **Title** - A summary of the problem.
- **Version** - The name of the affected Windows product version.
- **Status** - The current status of the issue.
- **Originating KB** - The KB number where the issue was first identified.
- **Originating build** - The build number for the KB.

Select the **Issue title** to access more information, including a link to the history of all status updates posted while we work on a solution. For example:

Sign up for email notifications

You can sign up for email notifications about Windows known issues and informational updates. Notifications include changes in issue status, new workarounds, and issue resolutions. To subscribe to notifications:

1. Go to the Windows release health page.
2. Select **Preferences** > **Email**, then select **Send me email notifications about Windows release health**.
3. Specify the following information:

    - Email address for the notifications
        - Each admin account can specify up to two email addresses under their email preferences
    - Windows versions to be notified about

4. Select **Save** when you're finished specifying email addresses and Windows versions. It may take up to 8 hours for these changes to take effect.

> **Note**

When a single known issue affects multiple versions of Windows, you'll receive only one email notification, even if you've selected notifications for multiple versions. Duplicate emails won't be sent.

Working with the Windows updates API in Microsoft Graph

If you'd like to develop an alternative way to get information on known issues documented within the Windows release health section in the admin center, you can use the Windows updates API in Microsoft Graph.

The Windows updates API has current and historical known issues data for any supported Windows product. You can check if an issue is confirmed, and if a resolution is available before calling support or spending time troubleshooting.

The Windows updates API also has product lifecycle information. For instance, you can search for end of servicing dates for all supported Windows versions and editions you manage in your organization. For more information on how to access these known issue and lifecycle data, see Microsoft Graph product resource type.

> **Note**
>
> These Windows data sets are currently under the Microsoft Graph REST API beta endpoint reference.

Status definitions

In the **Windows release health** experience, every known issue is assigned as status. Those statuses are defined as follows:

Expand table

Status	Definition
Reported	An issue has been brought to the attention of the Windows teams. At this stage, there's no confirmation that users are affected.
Investigating	The issue is believed to affect users and efforts are underway to gather more information about the issue's scope, mitigation steps, and root cause.

Status	Definition
Confirmed	After close review, Microsoft has determined the issue is affecting Windows users, and progress is being made on mitigation steps and root cause.
Mitigated	A workaround is available and communicated to Windows customers for a known issue. A known issue stays in this state until a KB article is released by Microsoft to resolve the known issue.
Mitigated: External	A workaround is available and communicated to Windows customers for a known issue caused by a software or driver from a third-party software or device manufacturer. A known issue stays in this state until the issue is resolved by Microsoft or the third-party.
Resolved	A solution was released by Microsoft and was documented in a KB article that resolves the known issue once it's deployed in the customer's environment.
Resolved: External	A solution was released by Microsoft or a third-party that resolves the known issue once it's deployed in the customer's environment.

Known issue history

The Windows release health page lets you view the history of all status updates posted for a specific known issue. To view all past updates posted for a given issue, select **View history** on the issue detail page.

A list of all status updates posted in the selected time frame is displayed. You can expand any row to view the specific information provided in that status update.

History: Stop error when plugging in a Thunderbolt NVMe SSD

User impact: Devices using Thunderbolt SSDs may receive a stop error "DRIVER_VERIFIER_DMA_VIOLATION (e6)" when plugging an SSD in.

December 11, 2020 5:46 PM

An incompatibility issue has been found with Windows 10, version 2004 or Windows 10, version 20H2 when using an Thunderbolt NVMe Solid State Disk (SSD). On affected devices, when plugging in a Thunderbolt NVMe SSD you might receive a stop error with a blue screen and "DRIVER_VERIFIER_DMA_VIOLATION (e6) An illegal DMA operation was attempted by a driver being verified." Affected Windows 10 devices will have at least one Thunderbolt port and any currently available version of the driver file stornvme.sys.

To safeguard your update experience, we have applied a compatibility hold on Windows 10 devices with affected drivers from being offered Windows 10, version 2004 or Windows 10, version 20H2. If your organization is using Update Compliance [link], the safeguard IDs is 29991611.

Resolution: This issue was resolved in KB4586853 and the safeguard hold has been removed as of December 11, 2020. Please note, if there are no other safeguards that affect your device, it can take up to 48 hours before the update to Windows 10, version 20H2 or Windows 10, version 2004 is offered.

Affected platforms:
- Client: Windows 10, version 20H2; Windows 10, version 2004

November 30, 2020 2:56 PM

November 30, 2020 2:19 PM

Frequently asked questions

Windows release health coverage

- **What is Windows release health?** Windows release health is a Microsoft informational service created to keep licensed Windows customers aware of identified known issues and important announcements.
- **Microsoft 365 service health content is specific to my tenants and services. Is the content in Windows release health specific to my Windows environment?** Windows release health doesn't monitor user environments or collect customer environment information. In Windows release health, all known issue content across all supported Windows versions is published to all subscribed customers. Future iterations of the solution may target content based on customer location, industry, or Windows version.
- **Where do I find Windows release health?** After logging into Microsoft 365 admin center, expand the left-hand menu using **...Show All**, select **Health** to display the **Windows release health** menu option.
- **Is the Windows release health content published to Microsoft 365 admin center the same as the content on Windows release health on Microsoft Learn?** No. While the content is similar, you may see more issues and more technical details published to

Windows release health on Microsoft 365 admin center to better support the IT admin. For example, you'll find details to help you diagnose issues in your environment, steps to mitigate issues, and root cause analysis.

- **How often will content be updated?** To ensure Windows customers have important information as soon as possible, all major known issues are shared with Windows customers on both Microsoft Learn and the Microsoft 365 admin center. We may also update the details available for Windows release health in the Microsoft 365 admin center when we have additional details on workarounds, root cause, or other information to help you plan for updates and handle issues in your environment.
- **Can I share this content publicly or with other Windows customers?** Windows release health is provided to you as a licensed Windows customer and isn't to be shared publicly.
- **Is the content redundant? How is the content organized in the different tabs?** Windows release health provides three tabs. The landing **All versions** tab allows you to select a specific version of Windows. The **Known issues** tab shows the list of issues that are active or resolved in the past 30 days. The **History** tab shows a six-month history of known issues that have been resolved.
- **How do I find information for the versions of Windows I'm managing?** On the **All versions** tab, you can select any Windows version. This action takes you to the **Known issues** tab filtered for the version you selected. The **Known issues** tab provides the list of active known issues and the issues resolved in the last 30 days. This selection persists throughout your session until changed. From the **History** tab, you can view the list of resolved issues for that version. To change versions, use the filter in the tab.

Microsoft 365 Admin Center functions

- **How do I best search for issues impacting my environment?** You can search Microsoft 365 admin center pages using keywords. For Windows release health, go to the desired product page and search using KB numbers, build numbers, or keywords.
- **How do I add other Windows admins?** Using the left-hand menu, go to Users, then select the Active Users tab and follow the prompts to add a new user, or assign an existing user, to the role of **Service Support admin**.
- **Why can't I click to the KB article from the Known issues or History tabs?** Within the issue description, you'll find links to the KB articles. In the known issue and history tabs, the entire row is a clickable entry to the issue's Details pane.
- **Microsoft 365 admin center has a mobile app but I don't see Windows release health under the Health menu. Is this an open issue?** We're working to build the Windows

release health experience on mobile devices in a future release.

Help and support

- **What should I do if I have an issue with Windows that is not reported in Windows release health?** Seek assistance through Premier support, the Microsoft Support website, or connect with your normal channels for Windows support.
- **When reaching out to Support, they asked me for an advisory ID. What is this and where can it?** The advisory ID can be found in the upper left-hand corner of the known issue Details pane. To find it, select the known issue you're seeking help on, select the **Details** pane, and you'll find the ID under the issue title. The ID is the letters `WI` followed by a number, similar to `WI123456`.
- **How can I learn more about expanding my use of Microsoft 365 admin center?** For more information, see the Microsoft 365 admin center documentation.

Last updated on 08/25/2025

Windows Autopilot documentation

Windows Autopilot and Windows Autopilot device preparation is a collection of technologies used to set up and pre-configure new devices, getting them ready for productive use.

Windows Autopilot solutions

GET STARTED

Compare Windows Autopilot device preparation and Windows Autopilot

Understand Windows Autopilot device preparation

OVERVIEW

Overview of Windows Autopilot device preparation

Requirements

Reporting and monitoring

Understand Windows Autopilot

OVERVIEW

Overview of Windows Autopilot

Requirements

Device registration overview

Enrollment Status Page

Manually register devices

Tutorials

TUTORIAL

Windows Autopilot scenarios

Windows Autopilot device preparation scenarios

Windows deployment scenarios and tools

Article • 08/30/2024

To successfully deploy the Windows operating system and applications for your organization, it's important to understand the available tools to help with the process. This article covers the most commonly used tools for Windows 10 deployment.

Microsoft provides many tools, services, and solutions. These tools include Windows Deployment Services (WDS), the Volume Activation Management Tool (VAMT), the User State Migration Tool (USMT), Windows System Image Manager (Windows SIM), Windows Preinstallation Environment (Windows PE), and Windows Recovery Environment (Windows RE). These tools aren't a complete solution on their own. Combine these tools with solutions like Configuration Manager to get a complete deployment solution.

This article also covers the different types of reference images that can be built, and why reference images are beneficial for most organizations.

Windows Assessment and Deployment Kit

The Windows Assessment and Deployment Kit (Windows ADK) contains core assessment and deployment tools and technologies, including:

- Deployment Image Servicing and Management (DISM).
- Windows Configuration Designer.
- Windows System Image Manager (Windows SIM).
- User State Migration Tool (USMT).
- Volume Activation Management Tool (VAMT).
- Windows Preinstallation Environment (Windows PE).
- Windows Assessment Toolkit.
- Windows Performance Toolkit (WPT).

For more information, see the following articles:

- Download and install the Windows ADK.
- Windows ADK for Windows scenarios for IT Pros.
- Kits and tools overview.

Deployment Image Servicing and Management (DISM)

DISM is one of the deployment tools included in the Windows ADK. It's used for capturing, servicing, and deploying both boot images and operating system images.

DISM services online and offline images. For example, with DISM you can install the Microsoft .NET Framework while Windows is online, which means that you can start the installation in the running operating system. The `/LimitAccess` switch configures DISM to get the files only from a local source. For example:

Windows Command Prompt

```
Dism.exe /Online /Enable-Feature /FeatureName:NetFX3 /All
/Source:D:\Sources\SxS /LimitAccess
```

Windows PowerShell can be used in Windows for many of the functions done by **DISM.exe**. The equivalent command in Windows using PowerShell is:

PowerShell

```
Enable-WindowsOptionalFeature -Online -FeatureName NetFx3 -All
-Source D:\Sources\SxS -LimitAccess
```

For more information on DISM, see DISM technical reference.

User State Migration Tool (USMT)

USMT is a backup and restore tool that allows you to migrate user state, data, and settings from one installation to another. Microsoft Configuration Manager uses USMT as part of the operating system deployment process.

USMT includes several command-line tools, the most important of which are **ScanState** and **LoadState**:

- **ScanState.exe**: This tool performs the user-state backup.
- **LoadState.exe**: This tool performs the user-state restore.
- **UsmtUtils.exe**: This tool supplements the functionality in **ScanState.exe** and **LoadState.exe**.

In addition to these tools, there are also XML templates that manage which data is migrated. You can customize the templates, or create new ones, to manage the backup process at a high level of detail. USMT uses the following terms for its templates:

- **Migration templates**: The default templates in USMT.
- **Custom templates**: Custom templates that you create.

- **Config template**: An optional template called **Config.xml** which you can use to exclude or include components in a migration without modifying the other standard XML templates.

USMT supports capturing and restoring both data and settings from currently supported versions of Windows. It also supports migrating from a 32-bit operating system to a 64-bit operating system, but not the other way around. For example, you can use USMT to migrate from Windows 10 x86 to Windows 11 x64.

By default USMT migrates many settings, most of which are related to the user profile but also to Control Panel configurations, file types, and more. The default templates that are used in Windows deployments are **MigUser.xml** and **MigApp.xml**. These two default templates migrate the following data and settings:

- Folders from each profile, including those folders from user profiles, and shared and public profiles. For example, the following folders:
  - Documents.
  - Video.
  - Music.
  - Pictures.
  - Desktop.

- The following specific file types:

  `.accdb`, `.ch3`, `.csv`, `.dif`, `.doc*`, `.dot*`, `.dqy`, `.iqy`, `.mcw`, `.mdb*`, `.mpp`, `.one*`, `.oqy`, `.or6`, `.pot*`, `.ppa`, `.pps*`, `.ppt*`, `.pre`, `.pst`, `.pub`, `.qdf`, `.qel`, `.qph`, `.qsd`, `.rqy`, `.rtf`, `.scd`, `.sh3`, `.slk`, `.txt`, `.vl*`, `.vsd`, `.wk*`, `.wpd`, `.wps`, `.wq1`, `.wri`, `.xl*`, `.xla`, `.xlb`, `.xls*`

  > ⓘ **Note**
  >
  > - The asterisk (`*`) stands for zero or more characters.
  > - The OpenDocument extensions (`*.odt`, `*.odp`, `*.ods`) used by Microsoft Office applications aren't migrated by default.

- Operating system component settings.

- Application settings.

These settings are migrated by the default **MigUser.xml** and **MigApp.xml** templates. For more information, see What does USMT migrate? For more general information on USMT, see User State Migration Tool (USMT) overview.

Windows Configuration Designer

Windows Configuration Designer is a tool designed to assist with the creation of provisioning packages that can be used to dynamically configure a Windows device. This tool is useful for setting up new devices without the need for reimaging the device with a custom image.

For more information, see Provisioning packages for Windows.

Windows System Image Manager (Windows SIM)

Windows System Image Manager (Windows SIM) is an authoring tool for **Unattend.xml** files. Windows SIM isn't normally needed when using Microsoft Configuration Manager. Microsoft Configuration Manager automatically creates and updates the **Unattend.xml** file based on settings specified in the task sequence, primarily at the **Apply Windows Settings** task. The automation in Microsoft Configuration Manager greatly simplifies the overall process.

For more information, see Windows System Image Manager Technical Reference.

Volume Activation Management Tool (VAMT)

If not using Key Management Services (KMS), Multiple Activation Keys (MAKs) can be centrally managed with the Volume Activation Management Tool (VAMT). Use this tool to install and manage product keys throughout the organization. VAMT can also activate on behalf of clients without internet access, acting as a MAK proxy.

VAMT can also be used to create reports, switch from MAK to KMS, manage Active Directory-based activation, and manage Office volume activation. VAMT also supports PowerShell. For example, to get information from the VAMT database, enter:

PowerShell

```
Get-VamtProduct
```

For more information on the VAMT, see the following articles:

- Volume Activation Management Tool (VAMT).
- VAMT technical reference.

Windows Preinstallation Environment (Windows PE)

Windows PE is a "lite" version of Windows used as a deployment platform.

Windows PE is like any other operating system and it needs drivers. However, it doesn't need a full set of drivers. It only needs a minimalist set of drivers necessary to deploy Windows. Normally only network and storage drivers are needed. Windows PE already includes a set of drivers out of the box so most devices work without the need to add any additional drivers.

For more information on Windows PE, see Windows PE (WinPE).

Windows Recovery Environment

Windows Recovery Environment (Windows RE) is a diagnostics and recovery toolset included in currently supported versions of Windows. Windows RE is based on Windows PE. If needed, Windows RE can also be extended with custom tools. If a Windows fails to start and Windows RE is installed, an automatic failover into Windows RE occurs.

For more information on Windows RE, see Windows Recovery Environment.

Windows Deployment Services

The main functions of Windows Deployment Services (WDS) are:

- PXE boot support.
- Multicast.
- BitLocker Network Unlock.

The following considerations should be observed when using WDS for operating system deployment:

- WDS can be configured for stand-alone mode or for Active Directory integration. Active Directory integration mode is recommended in most scenarios.

- WDS has the capability to manage drivers. However, driver management through Microsoft Configuration Manager is more suitable for deployment due to its flexibility.

- WDS can pre-stage unknown devices as a known computer in Active Directory. However, Microsoft Configuration Manager also has the capability of staging unknown devices as known devices in it's database. In most scenarios, Microsoft Configuration Manager is a better solution for pre-staging devices since it allows greater control and management.

- Trivial File Transfer Protocol (TFTP) block size and windows size settings can be configured with WDS to increase performance and download speeds during PXE booting. However, although an increase in TFTP settings can increase performance and download speeds, it can also decrease reliability and cause failures, including a reduction of download speeds. There are many variables involved when determining TFTP settings, including networking equipment, network configuration, and device compatibility.

  For stand-alone WDS, TFTP settings can be configured in the WDS console under the **TFTP** tab in the properties of the WDS server. For Microsoft Configuration manager, see Customize the RamDisk TFTP block and window sizes on PXE-enabled distribution points.

Windows Server Update Services

WSUS is a server role in Windows Server that enables a local repository of Microsoft updates. The Microsoft Update can then be distributed from the WSUS server to devices in the organization's environment without having to go out to the public Microsoft Update site. WSUS offers approval control and reporting of update status in the environment.

For more information on WSUS, see the Windows Server Update Services (WSUS).

Unified Extensible Firmware Interface

Unified Extensible Firmware Interface (**UEFI**) is used to initialize and boot a device. It's the successor for BIOS, the method used for many years to initialize and boot a device.

This section will go over the advantages of UEFI over BIOS, how the two differ, and now it affects operating system deployment.

Introduction to UEFI

Although BIOS was used successfully on devices for many years, it has some limitations. For example:

- 16-bit code
- 1-MB address space
- Poor performance on ROM initialization
- MBR maximum bootable disk size of 2.2 TB

As the replacement to BIOS, UEFI has many features BIOS doesn't have. Windows can take advantage of many of these UEFI features. UEFI has the following features not available in BIOS:

- **Support for large disks** - UEFI requires a GUID Partition Table (GPT) based disk. GPT can support disks up to approximately 16.8 million TB in disk size. GPT also supports more than 100 primary disks.
- **Faster boot time** - UEFI replaces BIOS interrupt call INT 13h, improving boot time, especially when resuming from hibernate.
- **Multicast deployment** - UEFI firmware can use multicast directly when it boots up. With solutions such as WDS and Microsoft Configuration Manager, multicast support is only available by first booting into Windows PE. With UEFI, multicast can run directly from UEFI.
- **Compatibility with earlier BIOS** - Older devices with UEFI had a UEFI implementation that included a compatibility support module (CSM) that emulates BIOS. However, due to the current wide support of UEFI, modern devices generally don't have a CSM and therefore aren't backward compatible with BIOS. For example, Windows 11 and newer doesn't support BIOS so therefore only runs on modern devices that have UEFI.
- **CPU-independent architecture** - BIOS can run both 32-bit and 64-bit versions of firmware. However, all firmware device drivers on BIOS systems must be 16-bit. This limitation affects performance and only 64 KB of memory can be addressed. UEFI removes these limitations.
- **CPU-independent drivers** - On BIOS systems, PCI add-on cards must include a ROM that contains a separate driver for all supported CPU architectures. This limitation isn't needed for UEFI because UEFI has the ability to use EFI Byte Code (EBC) images. EBC images allow for a processor-independent device driver environment.
- **Flexible pre-operating system environment** - UEFI supports UEFI application that can run before the OS is loaded. UEFI applications allow many additional features such as diagnostics, automatic repairs, and the ability to call home to report errors.
- **Secure boot** - Currently supported versions of Windows use the UEFI firmware validation process, called secure boot. When secure boot is used, UEFI ensures that it launches only a verified operating system loader and that malware can't switch the boot loader.

Hardware support for UEFI

In regard to UEFI, hardware is divided into four device classes:

- **Class 0 devices.** Devices in this class are BIOS, or non-UEFI, devices.
- **Class 1 devices.** Devices in this class behave like a standard BIOS device, but they run EFI internally. They should be treated as normal BIOS-based machines. Class 1 devices use a CSM to emulate BIOS.
- **Class 2 devices.** Devices in this class have the capability to behave as either a BIOS device or as a UEFI device. The boot process or the configuration in the firmware of the device determines the mode. Class 2 devices use a CSM to emulate BIOS.
- **Class 3 devices.** The devices in this class are UEFI-only devices. They don't have backwards compatibility with BIOS. Devices in this class must run an operating system that supports UEFI. All currently supported versions of Windows support UEFI. Class 3 devices don't have a CSM to emulate BIOS.

In general, all modern devices are Class 3 devices. Class 0, Class 1, and Class 2 devices are older devices and are no longer manufactured.

Windows support for UEFI

- Windows 10 supports both x86 and x64 versions of UEFI.
- Windows 11 and newer only supports x64 versions of UEFI.
- UEFI doesn't support cross-platform boot.
  - UEFI x64 devices can only run a 64-bit operating system. Most modern UEFI devices are x64.
  - UEFI x86 devices can run only a 32-bit operating system. For Windows, only Windows 10 x86 supports UEFI x86. Windows 11 and newer doesn't support UEFI x86. Lack of UEFI x86 support in Windows 11 generally isn't an issue since UEFI x86 devices are rare.

UEFI considerations for operating system deployment

There are many things that affect operating system deployment as soon as you run on UEFI/EFI-based hardware. Here are considerations to keep in mind when working with UEFI devices:

- Class 2 devices can switch between BIOS and UEFI via the device's firmware. Make sure the desired mode for the device is selected in the device's firmware. Microsoft recommends using Class 2 devices in UEFI mode due to the added benefits and security that UEFI provides.

  When a Class 2 device is switched from BIOS to UEFI, one of the following two actions needs to take place:

  - The disk needs to be converted from MBR to GPT and then partitioned accordingly to support UEFI. This conversion can be done via a tool such as diskpart. For example, while Windows running on BIOS only requires one partition that can be either FAT32 or NTFS, Windows running on a UEFI device requires the following partitions:
    - FAT32 boot/system partition.
    - NTFS OS partition.
    - Microsoft reserved partition (MSR) partition (unique to Windows).
    - Recovery partition (optional).

    Because the existing disk is wiped as part of this process, the following actions need to take place:
    - Windows need to be reinstalled.
    - Applications need to be reinstalled.
    - Data and settings need to be backed up and restored.

    For more information, see UEFI/GPT-based hard drive partitions.

  - The MBR2GPT.EXE tool can be used to convert the disk from MBR to GPT for use with UEFI in a non-destructive way. **MBR2GPT.EXE** also reconfigures the partitioning on the disk with the correct partitioning for Windows to run on UEFI. The benefit of using the **MBR2GPT.EXE** is that it converts the disk and repartitions it without wiping the disk and without data loss. Since the disk isn't wiped and there's no data loss, the following actions don't need to be performed:
    - Windows doesn't need to be reinstalled.
    - Applications don't need to be reinstalled.
    - Data and settings don't need to be backed up and restored.

- When you deploy a Class 2 device, make sure the boot option is set to the proper boot device (hard drive, flash drive, PXE, etc.) The boot options available on Class 2 devices might differ between BIOS and UEFI modes.

- When a UEFI device boots from media, the media has to be FAT32. UEFI only supports booting from FAT32 partitions, which is why the boot/system partition on

the disk is FAT32. Additionally, FAT32 has a file-size limitation of 4 GB. OS images larger than 4 GB need to be split with a tool such as DISM. For more information, see Split-WindowsImage or /Split-Image.

- UEFI doesn't support cross architecture booting. x64 devices require x64 boot media and x86 devices require x86 boot media.
- Most modern UEFI devices are x64. UEFI x86 devices are rare.

For more information on UEFI, see the UEFI firmware overview and related resources.

Related content

- Windows ADK for Windows scenarios for IT Pros.
- MBR2GPT.EXE.
- UEFI/GPT-based hard drive partitions.
- UEFI firmware.

Feedback

Was this page helpful? Yes No

Provide product feedback

Assign devices to servicing channels for Windows updates

Article • 08/24/2023 • Applies to: ✅ Windows 11, ✅ Windows 10

> **Looking for consumer information?** See Windows Update: FAQ

Tip

If you're not familiar with the servicing or release channels, read Servicing Channels first.

The General Availability Channel is the default servicing channel for all Windows 10 and Windows 11 devices except devices with the LTSC edition installed. The following table shows the servicing channels available to each edition.

Expand table

Edition	General Availability Channel	Long-Term Servicing Channel	Insider Program
Home	No	No	Yes
Pro	Yes	No	Yes
Enterprise	Yes	No	Yes
Enterprise LTSC	No	Yes	No
Pro Education	Yes	No	Yes
Education	Yes	No	Yes

Note

The LTSC edition is only available through the Microsoft Volume Licensing Center.

Note

Devices will automatically receive updates from the General Availability Channel, unless they are configured to receive preview updates through the Windows Insider Program.

Enroll devices in the Windows Insider Program

To get started with the Windows Insider Program for Business, follow these steps:

1. On the Windows Insider website, select **Register** to register your organizational Microsoft Entra account.
2. Follow the prompts to register your tenant.
   Note: The signed-in user needs to be a **Global Administrator** of the Microsoft Entra domain in order to be able to register.
3. Make sure the **Allow Telemetry** setting is set to **2** or higher.
4. For Windows devices, set policies to manage preview builds and their delivery:

The **Manage preview builds** setting gives administrators control over enabling or disabling preview build installation on a device. You can also decide to stop preview builds once the release is public.

- Group Policy: **Computer Configuration/Administrative Templates/Windows Components/Windows Update/Windows Update for Business** - *Manage preview builds*
- MDM: **Update/ManagePreviewBuilds**

The **Branch Readiness Level** settings allow you to choose between preview flight rings, and allows you to defer or pause the delivery of updates.

- Group Policy: **Computer Configuration/Administrative Templates/Windows Components/Windows Update/ Windows Update for Business** - *Select when Preview Builds and feature updates are received*
- MDM: **Update/BranchReadinessLevel**

For more information, see Windows Insider Program for Business.

Block access to Windows Insider Program

To prevent devices in your organization from being enrolled in the Insider Program for early releases of Windows 10:

- Group Policy: Computer Configuration\Administrative Templates\Windows Components\Data Collection and Preview Builds**Toggle user control over Insider**

builds

- MDM: Policy CSP - System/AllowBuildPreview

> **Important**
>
> Starting with Windows 10, version 1709, this policy is replaced by **Manage preview builds** policy.
>
> - Group Policy: **Computer Configuration/Administrative Templates/Windows Components/Windows Update/Windows Update for Business** - *Manage preview builds*
> - MDM: **Update/ManagePreviewBuilds**

Feedback

Was this page helpful? Yes No

Provide product feedback

Deploy Windows client updates using Windows Server Update Services (WSUS)

Applies to: ✅ Windows 11, ✅ Windows 10, ✅ WSUS

> **Looking for consumer information?** See Windows Update: FAQ

WSUS is a Windows Server role available in the Windows Server operating systems. It provides a single hub for Windows updates within an organization. WSUS allows companies not only to defer updates but also to selectively approve them, choose when they're delivered, and determine which individual devices or groups of devices receive them. WSUS provides additional control over Windows Update client policies but doesn't provide all the scheduling options and deployment flexibility that Microsoft Configuration Manager provides.

When you choose WSUS as your source for Windows updates, you use Group Policy to point Windows client devices to the WSUS server for their updates. From there, updates are periodically downloaded to the WSUS server and managed, approved, and deployed through the WSUS administration console or Group Policy, streamlining enterprise update management. If you're currently using WSUS to manage Windows updates in your environment, you can continue to do so in Windows 11.

> **Note**
>
> WSUS is deprecated and is no longer adding new features. However, it continues to be supported for production deployments, and receives security and quality updates as per the product lifecycle. For more info, see Features removed or no longer developed in Windows Server.

Requirements for Windows client servicing with WSUS

To be able to use WSUS to manage and deploy Windows feature updates, you must use a supported WSUS on a supported operating system version:

- WSUS role in Windows Server 2016
- WSUS role in Windows Server 2019
- WSUS role in Windows Server 2022
- WSUS role in Windows Server 2025

For more information about deploying the WSUS role, see Windows Server Update Services (WSUS) overview.

WSUS scalability

To use WSUS to manage all Windows updates, some organizations may need access to WSUS from a perimeter network, or they might have some other complex scenario. WSUS is highly scalable and configurable for organizations of any size or site layout. For specific information about scaling WSUS, including upstream and downstream server configuration, branch offices, WSUS load balancing, and other complex scenarios, see Deploy Windows Server Update Services.

Configure automatic updates and update service location

When using WSUS to manage updates on Windows client devices, start by configuring the **Configure Automatic Updates** and **Intranet Microsoft Update Service Location** Group Policy settings for your environment. Doing so forces the affected clients to contact the WSUS server so that it can manage them. The following process describes how to specify these settings and deploy them to all devices in the domain.

To configure the Configure Automatic Updates and Intranet Microsoft Update Service Location Group Policy settings for your environment

1. Open Group Policy Management Console (gpmc.msc).
2. Expand *Forest\Domains\Your_Domain*.
3. Right-click **Your_Domain**, and then select **Create a GPO in this domain, and Link it here**.

> ⓘ **Note**
>
> In this example, the **Configure Automatic Updates** and **Intranet Microsoft Update Service Location** Group Policy settings are specified for the entire domain. This isn't a requirement; you can target these settings to any security group by using Security Filtering or a specific OU.

4. In the **New GPO** dialog box, name the new GPO **WSUS - Auto Updates and Intranet Update Service Location**.

5. Right-click the **WSUS - Auto Updates and Intranet Update Service Location** GPO, and then select **Edit**.

6. In the Group Policy Management Editor, go to Computer Configuration\Policies\Administrative Templates\Windows Components\Windows Update.

7. Right-click the **Configure Automatic Updates** setting, and then select **Edit**.

8. In the **Configure Automatic Updates** dialog box, select **Enable**.

9. Under **Options**, from the **Configure automatic updating** list, select **3 - Auto download and notify for install**, and then select **OK**.

> **ⓘ Important**
>
> - Use Regedit.exe to check that the following key isn't enabled, because it can break Windows Store connectivity: Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\WindowsUpdate\DoNotConnectToWindowsUpdateInternetLocations

- There are three other settings for automatic update download and installation dates and times. This is simply the option this example uses. For more examples of how to control automatic updates and other related policies, see [Configure Automatic Updates by Using Group Policy](#).

10. Right-click the **Specify intranet Microsoft update service location** setting, and then select **Edit**.

11. In the **Specify intranet Microsoft update service location** dialog box, select **Enable**.

12. Under **Options**, in the **Set the intranet update service for detecting updates** and **Set the intranet statistics server** options, type `http://Your_WSUS_Server_FQDN:PortNumber`, and then select **OK**.

> ⓘ **Note**
>
> - The URL `http://Your_WSUS_Server_FQDN:PortNumber` in the following image is just an example. In your environment, be sure to use the server name and port number for your WSUS instance.

- The default HTTP port for WSUS is 8530, and the default HTTP over Secure Sockets Layer (HTTPS) port is 8531. (The other options are 80 and 443; no other ports are supported.)

As Windows clients refresh their computer policies (the default Group Policy refresh setting is 90 minutes and when a computer restarts), computers start to appear in WSUS. Now that clients are communicating with the WSUS server, create the computer groups that align with your deployment rings.

Create computer groups in the WSUS Administration Console

You can use computer groups to target a subset of devices that have specific quality and feature updates. These groups represent your deployment rings, as controlled by WSUS. You can populate the groups either manually by using the WSUS Administration Console or automatically through Group Policy. Regardless of the method you choose, you must first create the groups in the WSUS Administration Console.

To create computer groups in the WSUS Administration Console

1. Open the WSUS Administration Console.

2. Go to *Server_Name*\Computers\All Computers, and then select **Add Computer Group**.

3. Type **Ring 2 Pilot Business Users** for the name, and then select **Add**.

4. Repeat these steps for the **Ring 3 Broad IT** and **Ring 4 Broad Business Users** groups. When you're finished, there should be three deployment ring groups.

Now that the groups have been created, add the computers to the computer groups that align with the desired deployment rings. You can do this through Group Policy or manually by using the WSUS Administration Console.

Use the WSUS Administration Console to populate deployment rings

Adding computers to computer groups in the WSUS Administration Console is simple, but it could take much longer than managing membership through Group Policy, especially if you have many computers to add. Adding computers to computer groups in the WSUS Administration Console is called *server-side targeting*.

In this example, you add computers to computer groups in two different ways: by manually assigning unassigned computers and by searching for multiple computers.

Manually assign unassigned computers to groups

When new computers communicate with WSUS, they appear in the **Unassigned Computers** group. From there, you can use the following procedure to add computers to their correct groups. For these examples, you use two Windows 10 PCs (WIN10-PC1 and WIN10-PC2) to add to the computer groups.

To assign computers manually

1. In the WSUS Administration Console, go to *Server_Name*\Computers\All Computers\Unassigned Computers.

   Here, you see the new computers that have received the GPO you created in the previous section and started communicating with WSUS. This example has only two computers; depending on how broadly you deployed your policy, you'll likely have many computers here.

2. Select both computers, right-click the selection, and then select **Change Membership**.

3. In the **Set Computer Group Membership** dialog box, select the **Ring 2 Pilot Business Users** deployment ring, and then select **OK**.

Because they were assigned to a group, the computers are no longer in the **Unassigned Computers** group. If you select the **Ring 2 Pilot Business Users** computer group, you'll see both computers there.

Search for multiple computers to add to groups

Another way to add multiple computers to a deployment ring in the WSUS Administration Console is to use the search feature.

To search for multiple computers

1. In the WSUS Administration Console, go to *Server_Name*\Computers\All Computers, right-click **All Computers**, and then select **Search**.

2. In the search box, type **WIN10**.

3. In the search results, select the computers, right-click the selection, and then select **Change Membership**.

4. Select the **Ring 3 Broad IT** deployment ring, and then select **OK**.

You can now see these computers in the **Ring 3 Broad IT** computer group.

Use Group Policy to populate deployment rings

The WSUS Administration Console provides a friendly interface from which you can manage Windows quality and feature updates. When you need to add many computers to their correct WSUS deployment ring, however, it can be time-consuming to do so manually in the WSUS Administration Console. For these cases, consider using Group Policy to target the correct computers, automatically adding them to the correct WSUS deployment ring based on an Active Directory security group. This process is called *client-side targeting*. Before enabling client-side targeting in Group Policy, you must configure WSUS to accept Group Policy computer assignment.

To configure WSUS to allow client-side targeting from Group Policy

1. Open the WSUS Administration Console, and go to *Server_Name*\Options, and then select **Computers**.

2. In the **Computers** dialog box, select **Use Group Policy or registry settings on computers**, and then select **OK**.

> **Note**
>
> This option is exclusively either-or. When you enable WSUS to use Group Policy for group assignment, you can no longer manually add computers through the WSUS Administration Console until you change the option back.

Now that WSUS is ready for client-side targeting, complete the following steps to use Group Policy to configure client-side targeting:

To configure client-side targeting

> **Tip**
>
> When using client-side targeting, consider giving security groups the same names as your deployment rings. Doing so simplifies the policy-creation process and helps ensure that you don't add computers to the incorrect rings.

1. Open Group Policy Management Console (gpmc.msc).

2. Expand Forest\Domains*Your_Domain*.

3. Right-click *Your_Domain*, and then select **Create a GPO in this domain, and Link it here**.

4. In the **New GPO** dialog box, type **WSUS - Client Targeting - Ring 4 Broad Business Users** for the name of the new GPO.

5. Right-click the **WSUS - Client Targeting - Ring 4 Broad Business Users** GPO, and then select **Edit**.

6. In the Group Policy Management Editor, go to Computer Configuration\Policies\Administrative Templates\Windows Components\Windows Update.

7. Right-click **Enable client-side targeting**, and then select **Edit**.

8. In the **Enable client-side targeting** dialog box, select **Enable**.

9. In the **Target group name for this computer** box, type *Ring 4 Broad Business Users*. This is the name of the deployment ring in WSUS to which these computers will be added.

> ⚠ **Warning**
>
> The target group name must match the computer group name.

10. Close the Group Policy Management Editor.

Now you're ready to deploy this GPO to the correct computer security group for the **Ring 4 Broad Business Users** deployment ring.

To scope the GPO to a group

1. In GPMC, select the **WSUS - Client Targeting - Ring 4 Broad Business Users** policy.
2. Select the **Scope** tab.
3. Under **Security Filtering**, remove the default **AUTHENTICATED USERS** security group, and then add the **Ring 4 Broad Business Users** group.

The next time the clients in the **Ring 4 Broad Business Users** security group receive their computer policy and contact WSUS, they'll be added to the **Ring 4 Broad Business Users** deployment ring.

Automatically approve and deploy feature updates

For clients that should have their feature updates approved as soon as they're available, you can configure Automatic Approval rules in WSUS.

> ⓘ **Note**
>
> WSUS respects the client device's servicing branch. If you approve a feature update while it's still in one branch, such as Insider Preview, WSUS will install the update only on devices that are in that servicing branch. When Microsoft releases the build for the General Availability Channel, the devices in that will install it. Windows Update client policies branch settings don't apply to feature updates through WSUS.

To configure an Automatic Approval rule for Windows client feature updates and approve them for the Ring 3 Broad IT deployment ring This example uses Windows 10, but the process is the same for Windows 11.

1. In the WSUS Administration Console, go to Update Services*Server_Name*\Options, and then select **Automatic Approvals**.

2. On the **Update Rules** tab, select **New Rule**.

3. In the **Add Rule** dialog box, select the **When an update is in a specific classification**, **When an update is in a specific product**, and **Set a deadline for the approval** check boxes.

4. In the **Edit the properties** area, select **any classification**. Clear everything except **Upgrades**, and then select **OK**.

5. In the **Edit the properties area**, select the **any product** link. Clear all check boxes except **Windows 10**, and then select **OK**.

   Windows 10 is under All Products\Microsoft\Windows.

6. In the **Edit the properties** area, select the **all computers** link. Clear all the computer group check boxes except **Ring 3 Broad IT**, and then select **OK**.

7. Leave the deadline set for **7 days after the approval at 3:00 AM**.

8. In the **Step 3: Specify a name** box, type **Windows 10 Upgrade Auto-approval for Ring 3 Broad IT**, and then select **OK**.

9. In the **Automatic Approvals** dialog box, select **OK**.

> ⓘ **Note**
>
> WSUS doesn't honor any existing month/week/day deferral settings. That said, if you're using Windows Update client policies for a computer for which WSUS is also managing updates, when WSUS approves the update, it will be installed on the computer regardless of whether you configured Group Policy to wait.

Now, whenever Windows client feature updates are published to WSUS, they'll automatically be approved for the **Ring 3 Broad IT** deployment ring with an installation deadline of 1 week.

> ⚠ **Warning**
>
> The auto approval rule runs after synchronization occurs. This means that the *next* upgrade for each Windows client version will be approved. If you select **Run Rule**, all possible updates that meet the criteria will be approved, potentially including older updates that you don't actually want--which can be a problem when the download sizes are very large.

Manually approve and deploy feature updates

You can manually approve updates and set deadlines for installation within the WSUS Administration Console, as well. It might be best to approve update rules manually after your pilot deployment has been updated.

To simplify the manual approval process, start by creating a software update view that contains only Windows 10 (in this example) updates. The process is the same for Windows 11 updates.

> **Note**
>
> If you approve more than one feature update for a computer, an error can result with the client. Approve only one feature update per computer.

To approve and deploy feature updates manually

1. In the WSUS Administration Console, go to Update Services*Server_Name*\Updates. In the **Action** pane, select **New Update View**.

2. In the **Add Update View** dialog box, select **Updates are in a specific classification** and **Updates are for a specific product**.

3. Under **Step 2: Edit the properties**, select **any classification**. Clear all check boxes except **Upgrades**, and then select **OK**.

4. Under **Step 2: Edit the properties**, select **any product**. Clear all check boxes except **Windows 10**, and then select **OK**.

   Windows 10 is under All Products\Microsoft\Windows.

5. In the **Step 3: Specify a name** box, type **All Windows 10 Upgrades**, and then select **OK**.

Now that you have the **All Windows 10 Upgrades** view, complete the following steps to manually approve an update for the **Ring 4 Broad Business Users** deployment ring:

1. In the WSUS Administration Console, go to Update Services*Server_Name*\Updates\All Windows 10 Upgrades.

2. Right-click the feature update you want to deploy, and then select **Approve**.

3. In the **Approve Updates** dialog box, from the **Ring 4 Broad Business Users** list, select **Approved for Install**.

4. In the **Approve Updates** dialog box, from the **Ring 4 Broad Business Users** list, select **Deadline**, select **One Week**, and then select **OK**.

5. If the **Microsoft Software License Terms** dialog box opens, select **Accept**.

   If the deployment is successful, you should receive a successful progress report.

6. In the **Approval Progress** dialog box, select **Close**.

Last updated on 06/17/2025

Update Windows installation media with Dynamic Update

Applies to: ✅ Windows 11, ✅ Windows 10, ✅ Windows Server

This article explains how to acquire and apply Dynamic Update packages to existing Windows images *prior to deployment* and includes Windows PowerShell scripts you can use to automate this process.

Volume-licensed media is available for each release of Windows in the Volume Licensing Service Center (VLSC) and other relevant channels such as Windows Update client policies, Windows Server Update Services (WSUS), and Visual Studio Subscriptions. You can use Dynamic Update to ensure that Windows devices have the latest feature update packages as part of an in-place upgrade while preserving language pack and Features on Demand (FODs) that might have been previously installed. Dynamic Update also eliminates the need to install a separate quality update as part of the in-place upgrade process.

Dynamic Update

Whenever installation of a feature update starts (whether from media or an environment connected to Windows Update), *Dynamic Update* is one of the first steps. Windows Setup contacts a Microsoft endpoint to fetch Dynamic Update packages, and then applies those updates to your operating system installation media. The update packages include the following kinds of updates:

- Updates to Setup.exe binaries or other files that Setup uses for feature updates
- Updates for the "safe operating system" (SafeOS) that is used for the Windows recovery environment
- Updates to the servicing stack necessary to complete the feature update For more information, see Servicing stack updates.
- The latest cumulative (quality) update
- Updates to applicable drivers already published by manufacturers specifically intended for Dynamic Update

Dynamic Update preserves language pack and Features on Demand packages by reacquiring them.

Devices must be able to connect to the internet to obtain Dynamic Updates. In some environments, it's not an option to obtain Dynamic Updates. You can still do a media-based feature update by acquiring Dynamic Update packages and applying it to the image prior to starting Setup on the device.

Acquire Dynamic Update packages

You can obtain Dynamic Update packages from the Microsoft Update Catalog. At that site, use the search bar in the upper right to find the Dynamic Update packages for a particular release. The various Dynamic Update packages might not all be present in the results from a single search, so you might have to search with different keywords to find all of the updates. Check various parts of the results to be sure you've identified the files needed. The following tables show the key values to search for or look for in the results.

Windows Server 2025 Dynamic Update packages

Title can distinguish each Dynamic Package. Latest cumulative updates have the servicing stack embedded. The servicing stack is published only if necessary for a given cumulative update.

Expand table

Update packages	Title
Safe OS Dynamic Update	YYYY-MM Safe OS Dynamic Update for Microsoft server operating system version 24H2
Setup Dynamic Update	YYYY-MM Setup Dynamic Update for Microsoft server operating system version 24H2
Latest cumulative update	YYYY-MM Cumulative Update for Microsoft server operating system version 24H2
Servicing stack Dynamic Update	YYYY-MM Servicing Stack Update for Microsoft server operating system version 24H2

Windows Server, version 23H2 Dynamic Update packages

Title can distinguish each Dynamic Package. Latest cumulative updates have the servicing stack embedded. The servicing stack is published only if necessary for a given cumulative update. Azure Stack HCI, version 23H2 has a similar format.

Expand table

Update packages	Title
Safe OS Dynamic Update	YYYY-MM Safe OS Dynamic Update for Microsoft server operating system version 23H2
Setup Dynamic Update	YYYY-MM Setup Dynamic Update for Microsoft server operating system

Update packages	Title
	version 23H2
Latest cumulative update	YYYY-MM Cumulative Update for Microsoft server operating system version 23H2
Servicing stack Dynamic Update	YYYY-MM Servicing Stack Update for Microsoft server operating system version 23H2

Azure Stack HCI, version 22H2 Dynamic Update packages

Title, **Product**, and **Description** are required to distinguish each Dynamic Package. Latest cumulative update has the servicing stack embedded. Servicing stack published separately only if necessary as a prerequisite for a given cumulative update.

Expand table

Update packages	Title	Product	Description
Safe OS Dynamic Update	YYYY-MM Dynamic Update for Microsoft server operating system, version 22H2	Windows Safe OS Dynamic Update	ComponentUpdate
Setup Dynamic Update	YYYY-MM Dynamic Update for Microsoft server operating system, version 22H2	Windows 10 and later Dynamic Update	SetupUpdate
Latest cumulative update	YYYY-MM Cumulative Update for Microsoft server operating system, version 22H2		
Servicing stack Dynamic Update	YYYY-MM Servicing Stack Update for Microsoft server operating system, version 22H2		

Windows Server 2022 later Dynamic Update packages

Title, **Product**, and **Description** are required to distinguish each Dynamic Package. Latest cumulative update has the servicing stack embedded. Servicing stack published separately only if necessary as a prerequisite for a given cumulative update.

Expand table

Update packages	Title	Product	Description
Safe OS Dynamic Update	YYYY-MM Dynamic Update for Microsoft server operating system, version 21H2	Windows Safe OS Dynamic Update	ComponentUpdate
Setup Dynamic Update	YYYY-MM Dynamic Update for Microsoft server operating system, version 21H2	Windows 10 and later Dynamic Update	SetupUpdate
Latest cumulative update	YYYY-MM Cumulative Update for Microsoft server operating system, version 21H2		
Servicing stack Dynamic Update	YYYY-MM Servicing Stack Update for Microsoft server operating system, version 21H2		

Windows 11, version 22H2, and later Dynamic Update packages

Title can distinguish each Dynamic Package. Latest cumulative updates have the servicing stack embedded. The servicing stack is published only if necessary for a given cumulative update. The following titles are for Windows 11, version 22H2. Windows 11, version 23H2, and version 24H2 have a similar format:

Expand table

Update packages	Title
Safe OS Dynamic Update	YYYY-MM Safe OS Dynamic Update for Windows 11 Version 22H2
Setup Dynamic Update	YYYY-MM Setup Dynamic Update for Windows 11 Version 22H2
Latest cumulative update	YYYY-MM Cumulative Update for Windows 11 Version 22H2
Servicing stack Dynamic Update	YYYY-MM Servicing Stack Update for Windows 11 Version 22H2

Windows 11, version 21H2 Dynamic Update packages

Title, **Product**, and **Description** are required to distinguish each Dynamic Package. Latest cumulative update has the servicing stack embedded. Servicing stack published separately only if necessary as a prerequisite for a given cumulative update.

Expand table

Update packages	Title	Product	Description
Safe OS Dynamic Update	YYYY-MM Dynamic Update for Windows 11	Windows Safe OS Dynamic Update	ComponentUpdate
Setup Dynamic Update	YYYY-MM Dynamic Update for Windows 11	Windows 10 and later Dynamic Update	SetupUpdate
Latest cumulative update	YYYY-MM Cumulative Update for Windows 11		
Servicing stack Dynamic Update	YYYY-MM Servicing Stack Update for Windows 11 Version 21H2		

Windows 10, version 22H2 Dynamic Update packages

Title, **Product**, and **Description** are required to distinguish each Dynamic Package. Latest cumulative update has the servicing stack embedded. Servicing stack published separately only if necessary as a prerequisite for a given cumulative update.

Expand table

Update packages	Title	Product	Description
Safe OS Dynamic Update	YYYY-MM Dynamic Update for Windows 10 Version 22H2	Windows Safe OS Dynamic Update	ComponentUpdate
Setup Dynamic Update	YYYY-MM Dynamic Update for Windows 10 Version 22H2	Windows 10 and later Dynamic Update	SetupUpdate
Latest cumulative update	YYYY-MM Cumulative Update for Windows 10 Version 22H2		
Servicing stack Dynamic Update	YYYY-MM Servicing Stack Update for Windows 10 Version 22H2		

If you want to customize the image with additional languages or Features on Demand, download supplemental media ISO files from the Volume Licensing Service Center. For example, if Dynamic Update will be disabled for your devices, and if users require specific Features on Demand, you can preinstall these into the image.

Update Windows installation media

Properly updating the installation media involves many actions operating on several different targets (image files). Some actions are repeated on different targets. The target images files include:

- Windows Preinstallation Environment (WinPE): a small operating system used to install, deploy, and repair Windows operating systems
- Windows Recovery Environment (WinRE): repairs common causes of unbootable operating systems. WinRE is based on WinPE and can be customized with additional drivers, languages, optional packages, and other troubleshooting or diagnostic tools.
- Windows operating system: one or more editions of Windows stored in \sources\install.wim
- Windows installation media: the complete collection of files and folders in the Windows installation media. For example, \sources folder, \boot folder, Setup.exe, and so on.

This table shows the correct sequence for applying the various tasks to the files. For example, the full sequence starts with adding the servicing stack update to WinRE (1) and concludes with adding boot manager from WinPE to the new media (28).

Expand table

Task	WinRE (winre.wim)	Operating system (install.wim)	WinPE (boot.wim)	New media
Add servicing stack update via latest cumulative update	1	9	17	
Add language pack	2	10	18	
Add localized optional packages	3		19	
Add font support	4		20	
Add text-to-speech	5		21	
Update Lang.ini			22	
Add Features on Demand		11		
Add Optional Components		12		
Add Safe OS Dynamic Update	6			
Add Setup Dynamic Update				26
Add Setup.exe and setuphost.exe from WinPE				27
Add boot manager from WinPE				28
Add latest cumulative update		13	23	
Clean up the image	7	14	24	

Task	WinRE (winre.wim)	Operating system (install.wim)	WinPE (boot.wim)	New media
Add .NET and .NET cumulative updates		15		
Export image	8	16	25	

> **Note**
>
> - Starting with Windows 11, version 24H2, and Windows Server 2025, the latest cumulative update might have a prerequisite cumulative update that is required to be installed first. See the **Checkpoint cumulative updates** section below.
> - Starting in February 2021, the latest cumulative update and servicing stack update is combined and distributed in the Microsoft Update Catalog as a new combined cumulative update. For Steps 1, 9, and 17 that require the servicing stack update for updating the installation media, you should use the combined cumulative update. For more information on the combined cumulative update, see Servicing stack updates. When acquiring Dynamic Update packages, ensure the packages correspond to the same month as the latest cumulative update. If, for example, the SafeOS Dynamic Update or Setup Dynamic Update is not available for the same month as the latest cumulative update, use the most recent published version of each.
> - Microsoft removes the Flash component from Windows through KB4577586: Update for Removal of Adobe Flash Player. You can also remove Flash anytime by deploying the update in KB4577586 (available on the Catalog) between steps 20 and 21. As of July 2021, KB4577586, "Update for Removal of Adobe Flash Player" will be included in the latest cumulative update for Windows 10, versions 1607 and 1507. The update will also be included in the Monthly Rollup and the Security Only Update for Windows 8.1, Windows Server 2012, and Windows Embedded 8 Standard. For more information, see Update on Adobe Flash Player End of Support.

Multiple Windows editions

The main operating system file (install.wim) might contain multiple editions of Windows. It's possible that only an update for a given edition is required to deploy it, based on the index. Or, it might be that all editions need an update. Further, ensure that languages are installed before Features on Demand, and the latest cumulative update is always applied last.

Additional languages and features

You don't have to add more languages and features to the image to accomplish the updates, but it's an opportunity to customize the image with more languages, Optional Components, and Features on Demand beyond what's in your starting image. When you add more languages and features, it's important to make these changes in the correct order: first apply servicing stack updates, followed by language additions, then by feature additions, and finally the latest cumulative update. The provided sample script installs a second language (in this case Japanese (ja-JP)). Since this language is backed by an lp.cab, there's no need to add a Language Experience Pack. Japanese is added to both the main operating system and to the recovery environment to allow the user to see the recovery screens in Japanese. This includes adding localized versions of the packages currently installed in the recovery image.

Optional Components, along with the .NET feature, can be installed offline. However, doing so creates pending operations that require the device to restart. As a result, the call to perform image cleanup would fail. There are two options to avoid the cleanup failure. One option is to skip the image cleanup step, though that results in a larger install.wim. Another option is to install the .NET and Optional Components in a step after cleanup but before export. This is the option in the sample script. By doing this, you'll have to start with the original install.wim (with no pending actions) when you maintain or update the image the next time (for example, the next month).

Checkpoint cumulative updates

Starting with Windows 11, version 24H2, and Windows Server 2025, the latest cumulative update might have a prerequisite cumulative update that is required to be installed first. These updates are known as checkpoint cumulative updates. In these cases, the cumulative update file level differentials are based on a previous cumulative update instead of the Windows RTM release. The benefit is a smaller update package and faster installation. When you obtain the latest cumulative update from the Microsoft Update Catalog, checkpoint cumulative updates are available from the download button. In addition, the knowledge base article for the cumulative update provides additional information.

To install the checkpoint(s) when servicing the Windows OS (steps 9 & 12) and WinPE (steps 17 & 23), call `Add-WindowsPackage` with the target cumulative update. The folder from `-PackagePath` is used to discover and install one or more checkpoints as needed. Only the target cumulative update and checkpoint cumulative updates should be in the `-PackagePath` folder. Cumulative update packages with a revision <= the target cumulative update are processed. If you aren't customizing the image with additional languages and/or optional features, then separate calls to `Add-WindowsPackage` (checkpoint cumulative updates first) can be used for steps 9 & 17 above. Separate calls can't be used for steps 12 and 23.

Depending on the revision of the image you are servicing, as well as your specific image custimization task, you may or may not need one or all of the preceding checkpoints. Let's look at three examples:

Example 1: Adding a language or feature: If you are adding a language or feature to an existing image, the target cumulative update and all preceding checkpoints are required to be installed. This is required as languages and features are published only once when a new Windows OS was released to manufacturing (RTM), and thus require all servicing changes since RTM to avoid torn state.

Example 2: Updating an image on the latest checkpoint: If you are not adding a language or feature, and your existing image is on the latest checkpoint, then only the target cumulative update is needed to get current.

Example 3: Updating an image not on the latest checkpoint: If you are not adding a language or feature, and your existing image is not on the latest checkpoint, then one or more preceding checkpoints are required, along with the target cumulative update, to get current.

For more information, see Checkpoint cumulative updates and Microsoft Update Catalog usage.

Windows PowerShell scripts to apply Dynamic Updates to an existing image

These examples are for illustration only, and therefore lack error handling. The script assumes that the following packages are stored locally in this folder structure:

Expand table

Folder	Description
C:\mediaRefresh	Parent folder that contains the PowerShell script
C:\mediaRefresh\oldMedia	Folder that contains the original media that will be refreshed. For example, contains Setup.exe, and \sources folder.
C:\mediaRefresh\newMedia	Folder that will contain the updated media. It's copied from \oldMedia, then used as the target for all update and cleanup operations.

Get started

The script starts by declaring global variables and creating folders to use for mounting images. Then, make a copy of the original media, from \oldMedia to \newMedia, keeping the original

media in case there's a script error and it's necessary to start over from a known state. Also, it provides a comparison of old versus new media to evaluate changes. To ensure that the new media updates, make sure they aren't read-only. The script also showcases adding additional languages, Features on Demand, and Optional Components. These aren't required, but added to highlight when in the sequence they should be added. Starting with Windows 11, version 21H2, the language pack (LANGPACK) ISO is superseded by the Features on Demand ISO. Language packs and the \Windows Preinstallation Environment packages are part of the Features on Demand ISO. Further, the path for main OS language and optional features moved to \LanguagesAndOptionalFeatures instead of the root. If you're using this script for Windows 10, modify to mount and use the language pack (LANGPACK) ISO.

PowerShell

```
#Requires -RunAsAdministrator

function Get-TS { return "{0:HH:mm:ss}" -f [DateTime]::Now }

Write-Output "$(Get-TS): Starting media refresh"

# Declare Dynamic Update packages. A dedicated folder is used for the latest
cumulative update, and as needed
# checkpoint cumulative updates.
$LCU_PATH        = "C:\mediaRefresh\packages\CU\LCU.msu"
$SETUP_DU_PATH   = "C:\mediaRefresh\packages\Other\Setup_DU.cab"
$SAFE_OS_DU_PATH = "C:\mediaRefresh\packages\Other\SafeOS_DU.cab"
$DOTNET_CU_PATH  = "C:\mediaRefresh\packages\Other\DotNet_CU.msu"

# Declare media for FOD and LPs
$FOD_ISO_PATH    = "C:\mediaRefresh\packages\CLIENT_LOF_PACKAGES_OEM.iso"

# Array of Features On Demand for main OS
# This is optional to showcase where these are added
$FOD = @(
'XPS.Viewer~~~~0.0.1.0'
)

# Array of Legacy Features for main OS
# This is optional to showcase where these are added
$OC = @(
'MediaPlayback'
'WindowsMediaPlayer'
)

# Mount the Features on Demand ISO
Write-Output "$(Get-TS): Mounting FOD ISO"
$FOD_ISO_DRIVE_LETTER = (Mount-DiskImage -ImagePath $FOD_ISO_PATH -ErrorAction stop
| Get-Volume).DriveLetter
$FOD_PATH = $FOD_ISO_DRIVE_LETTER + ":\LanguagesAndOptionalFeatures"

# Declare language for showcasing adding optional localized components
$LANG  = "ja-jp"
```

```
$LANG_FONT_CAPABILITY = "jpan"

# Declare language related cabs
$WINPE_OC_PATH              = "$FOD_ISO_DRIVE_LETTER`:\Windows Preinstallation
Environment\x64\WinPE_OCs"
$WINPE_OC_LANG_PATH         = "$WINPE_OC_PATH\$LANG"
$WINPE_OC_LANG_CABS         = Get-ChildItem $WINPE_OC_LANG_PATH -Name
$WINPE_OC_LP_PATH           = "$WINPE_OC_LANG_PATH\lp.cab"
$WINPE_FONT_SUPPORT_PATH    = "$WINPE_OC_PATH\WinPE-FontSupport-$LANG.cab"
$WINPE_SPEECH_TTS_PATH      = "$WINPE_OC_PATH\WinPE-Speech-TTS.cab"
$WINPE_SPEECH_TTS_LANG_PATH = "$WINPE_OC_PATH\WinPE-Speech-TTS-$LANG.cab"
$OS_LP_PATH                 = "$FOD_PATH\Microsoft-Windows-Client-Language-
Pack_x64_$LANG.cab"

# Declare folders for mounted images and temp files
$MEDIA_OLD_PATH  = "C:\mediaRefresh\oldMedia\Ge\client_professional_en-us"
$MEDIA_NEW_PATH  = "C:\mediaRefresh\newMedia"
$WORKING_PATH    = "C:\mediaRefresh\temp"
$MAIN_OS_MOUNT   = "C:\mediaRefresh\temp\MainOSMount"
$WINRE_MOUNT     = "C:\mediaRefresh\temp\WinREMount"
$WINPE_MOUNT     = "C:\mediaRefresh\temp\WinPEMount"

# Create folders for mounting images and storing temporary files
New-Item -ItemType directory -Path $WORKING_PATH -ErrorAction Stop | Out-Null
New-Item -ItemType directory -Path $MAIN_OS_MOUNT -ErrorAction stop | Out-Null
New-Item -ItemType directory -Path $WINRE_MOUNT -ErrorAction stop | Out-Null
New-Item -ItemType directory -Path $WINPE_MOUNT -ErrorAction stop | Out-Null

# Keep the original media, make a copy of it for the new, updated media.
Write-Output "$(Get-TS): Copying original media to new media path"
Copy-Item -Path $MEDIA_OLD_PATH"\*" -Destination $MEDIA_NEW_PATH -Force -Recurse -
ErrorAction stop | Out-Null
Get-ChildItem -Path $MEDIA_NEW_PATH -Recurse | Where-Object { -not $_.PSIsContainer
-and $_.IsReadOnly } | ForEach-Object { $_.IsReadOnly = $false }
```

Update WinRE and each main OS Windows edition

The script updates each edition of Windows within the main operating system file (install.wim). For each edition, the main OS image is mounted.

For the first image, Winre.wim is copied to the working folder, and mounted. It then applies servicing stack via the latest cumulative update, since its components are used for updating other components. Depending on the Windows release that you're updating, there are two different approaches for updating the servicing stack. The first approach is to use the combined cumulative update. This is for Windows releases that are shipping a combined cumulative update that includes the servicing stack updates (that is, SSU + LCU are combined). Windows 11, version 21H2, and Windows 11, version 22H2 are examples. In these cases, the servicing stack update isn't published separately; the combined cumulative update should be

used for this step. However, in rare cases, there might be a breaking change in the combined cumulative update format change, that requires a standalone servicing stack update to be published, and installed first before the combined cumulative update can be installed. Since the script is optionally adding Japanese, it adds the language pack to the image, and installs the Japanese versions of all optional packages already installed in Winre.wim. Then, it applies the Safe OS Dynamic Update package. It finishes by cleaning and exporting the image to reduce the image size.

Next, for the mounted OS image, the script starts by applying the servicing stack via the latest cumulative update. Then, it adds Japanese language support and then the Japanese language features. Unlike the Dynamic Update packages, it uses `Add-WindowsCapability` to add these features. For a full list of such features, and their associated capability name, see Available Features on Demand. Now is the time to enable other Optional Components or add other Features on Demand. If such a feature has an associated cumulative update (for example, .NET), this is the time to apply those. The script then attempts to clean the image, then a final step to apply the latest cumulative update. It's important to apply the latest cumulative update last, to ensure Features on Demand, Optional Components, and Languages are updated from their initial release state. The .NET feature is an exception that's added along with its cumulative update next. Finally, the script exports the image.

This process is repeated for each edition of Windows within the main operating system file. To reduce size, the serviced Winre.wim file from the first image is saved, and used to update each subsequent Windows edition. This reduces the final size of install.wim.

PowerShell

```
#
# Update each main OS Windows image including the Windows Recovery Environment
(WinRE)
#

# Get the list of images contained within the main OS
$WINOS_IMAGES = Get-WindowsImage -ImagePath $MEDIA_NEW_PATH"\sources\install.wim"

Foreach ($IMAGE in $WINOS_IMAGES)
{

    # first mount the main OS image
    Write-Output "$(Get-TS): Mounting main OS, image index $($IMAGE.ImageIndex)"
    Mount-WindowsImage -ImagePath $MEDIA_NEW_PATH"\sources\install.wim" -Index
$IMAGE.ImageIndex -Path $MAIN_OS_MOUNT -ErrorAction stop| Out-Null

    if ($IMAGE.ImageIndex -eq "1")
    {

        #
        # update Windows Recovery Environment (WinRE) within this OS image
```

```
        #
        Copy-Item -Path $MAIN_OS_MOUNT"\windows\system32\recovery\winre.wim" -
Destination $WORKING_PATH"\winre.wim" -Force -ErrorAction stop | Out-Null
        Write-Output "$(Get-TS): Mounting WinRE"
        Mount-WindowsImage -ImagePath $WORKING_PATH"\winre.wim" -Index 1 -Path
$WINRE_MOUNT -ErrorAction stop | Out-Null

        # Add servicing stack update (Step 1 from the table)
        Write-Output "$(Get-TS): Adding package $LCU_PATH to WinRE"
        try
        {
            Add-WindowsPackage -Path $WINRE_MOUNT -PackagePath $LCU_PATH | Out-Null
        }
        Catch
        {
            $theError = $_
            Write-Output "$(Get-TS): $theError"

            if ($theError.Exception -like "*0x8007007e*")
            {
                Write-Warning "$(Get-TS): Failed with error 0x8007007e. This failure
is a known issue with combined cumulative update, we can ignore."
            }
            else
            {
                throw
            }
        }

        #
        # Optional: Add the language to recovery environment
        #

        # Install lp.cab cab
        Write-Output "$(Get-TS): Adding package $WINPE_OC_LP_PATH to WinRE"
        Add-WindowsPackage -Path $WINRE_MOUNT -PackagePath $WINPE_OC_LP_PATH -
ErrorAction stop | Out-Null

        # Install language cabs for each optional package installed
        $WINRE_INSTALLED_OC = Get-WindowsPackage -Path $WINRE_MOUNT
        Foreach ($PACKAGE in $WINRE_INSTALLED_OC)
        {
            if ( ($PACKAGE.PackageState -eq "Installed") -and
($PACKAGE.PackageName.startsWith("WinPE-")) -and ($PACKAGE.ReleaseType -eq
"FeaturePack") )
            {
                $INDEX = $PACKAGE.PackageName.IndexOf("-Package")
                if ($INDEX -ge 0)
                {
                    $OC_CAB = $PACKAGE.PackageName.Substring(0, $INDEX) + "_" +
$LANG + ".cab"
                    if ($WINPE_OC_LANG_CABS.Contains($OC_CAB))
                    {
                        $OC_CAB_PATH = Join-Path $WINPE_OC_LANG_PATH $OC_CAB
                        Write-Output "$(Get-TS): Adding package $OC_CAB_PATH to
```

```
WinRE"
                        Add-WindowsPackage -Path $WINRE_MOUNT -PackagePath
$OC_CAB_PATH -ErrorAction stop | Out-Null
                    }
                }
            }
        }

        # Add font support for the new language
        if ( (Test-Path -Path $WINPE_FONT_SUPPORT_PATH) )
        {
            Write-Output "$(Get-TS): Adding package $WINPE_FONT_SUPPORT_PATH to
WinRE"
            Add-WindowsPackage -Path $WINRE_MOUNT -PackagePath
$WINPE_FONT_SUPPORT_PATH -ErrorAction stop | Out-Null
        }

        # Add TTS support for the new language
        if (Test-Path -Path $WINPE_SPEECH_TTS_PATH)
        {
            if ( (Test-Path -Path $WINPE_SPEECH_TTS_LANG_PATH) )
            {
                Write-Output "$(Get-TS): Adding package $WINPE_SPEECH_TTS_PATH to
WinRE"
                Add-WindowsPackage -Path $WINRE_MOUNT -PackagePath
$WINPE_SPEECH_TTS_PATH -ErrorAction stop | Out-Null

                Write-Output "$(Get-TS): Adding package $WINPE_SPEECH_TTS_LANG_PATH
to WinRE"
                Add-WindowsPackage -Path $WINRE_MOUNT -PackagePath
$WINPE_SPEECH_TTS_LANG_PATH -ErrorAction stop | Out-Null
            }
        }

        # Add Safe OS
        Write-Output "$(Get-TS): Adding package $SAFE_OS_DU_PATH to WinRE"
        Add-WindowsPackage -Path $WINRE_MOUNT -PackagePath $SAFE_OS_DU_PATH -
ErrorAction stop | Out-Null

        # Perform image cleanup
        Write-Output "$(Get-TS): Performing image cleanup on WinRE"
        DISM /image:$WINRE_MOUNT /cleanup-image /StartComponentCleanup /ResetBase
/Defer | Out-Null
        if ($LastExitCode -ne 0)
        {
            throw "Error: Failed to perform image cleanup on WinRE. Exit code:
$LastExitCode"
        }

        # Dismount
        Dismount-WindowsImage -Path $WINRE_MOUNT  -Save -ErrorAction stop | Out-Null

        # Export
        Write-Output "$(Get-TS): Exporting image to $WORKING_PATH\winre.wim"
        Export-WindowsImage -SourceImagePath $WORKING_PATH"\winre.wim" -SourceIndex
```

```
1 -DestinationImagePath $WORKING_PATH"\winre2.wim" -ErrorAction stop | Out-Null

    }

    Copy-Item -Path $WORKING_PATH"\winre2.wim" -Destination
$MAIN_OS_MOUNT"\windows\system32\recovery\winre.wim" -Force -ErrorAction stop | Out-
Null

    #
    # update Main OS
    #

    # Add servicing stack update (Step 17 from the table). Unlike WinRE and WinPE,
we don't need to check for error 0x8007007e
    Write-Output "$(Get-TS): Adding package $LCU_PATH to main OS, index
$($IMAGE.ImageIndex)"
    Add-WindowsPackage -Path $MAIN_OS_MOUNT -PackagePath $LCU_PATH | Out-Null


    # Optional: Add language to main OS and corresponding language experience
Features on Demand
    Write-Output "$(Get-TS): Adding package $OS_LP_PATH to main OS, index
$($IMAGE.ImageIndex)"
    Add-WindowsPackage -Path $MAIN_OS_MOUNT -PackagePath $OS_LP_PATH -ErrorAction
stop | Out-Null

    Write-Output "$(Get-TS): Adding language FOD: Language.Fonts.Jpan~~~und-
JPAN~0.0.1.0 to main OS, index $($IMAGE.ImageIndex)"
    Add-WindowsCapability -Name
"Language.Fonts.$LANG_FONT_CAPABILITY~~~und-$LANG_FONT_CAPABILITY~0.0.1.0" -Path
$MAIN_OS_MOUNT -Source $FOD_PATH -ErrorAction stop | Out-Null

    Write-Output "$(Get-TS): Adding language FOD: Language.Basic~~~$LANG~0.0.1.0 to
main OS, index $($IMAGE.ImageIndex)"
    Add-WindowsCapability -Name "Language.Basic~~~$LANG~0.0.1.0" -Path
$MAIN_OS_MOUNT -Source $FOD_PATH -ErrorAction stop | Out-Null

    Write-Output "$(Get-TS): Adding language FOD: Language.OCR~~~$LANG~0.0.1.0 to
main OS, index $($IMAGE.ImageIndex)"
    Add-WindowsCapability -Name "Language.OCR~~~$LANG~0.0.1.0" -Path $MAIN_OS_MOUNT
-Source $FOD_PATH -ErrorAction stop | Out-Null

    Write-Output "$(Get-TS): Adding language FOD:
Language.Handwriting~~~$LANG~0.0.1.0 to main OS, index $($IMAGE.ImageIndex)"
    Add-WindowsCapability -Name "Language.Handwriting~~~$LANG~0.0.1.0" -Path
$MAIN_OS_MOUNT -Source $FOD_PATH -ErrorAction stop | Out-Null

    Write-Output "$(Get-TS): Adding language FOD:
Language.TextToSpeech~~~$LANG~0.0.1.0 to main OS, index $($IMAGE.ImageIndex)"
    Add-WindowsCapability -Name "Language.TextToSpeech~~~$LANG~0.0.1.0" -Path
$MAIN_OS_MOUNT -Source $FOD_PATH -ErrorAction stop | Out-Null

    Write-Output "$(Get-TS): Adding language FOD: Language.Speech~~~$LANG~0.0.1.0 to
main OS, index $($IMAGE.ImageIndex)"
    Add-WindowsCapability -Name "Language.Speech~~~$LANG~0.0.1.0" -Path
```

```
$MAIN_OS_MOUNT -Source $FOD_PATH -ErrorAction stop | Out-Null

    # Optional: Add additional Features On Demand
    For ( $index = 0; $index -lt $FOD.count; $index++)#
    {
        Write-Output "$(Get-TS): Adding $($FOD[$index]) to main OS, index
$($IMAGE.ImageIndex)"
        Add-WindowsCapability -Name $($FOD[$index]) -Path $MAIN_OS_MOUNT -Source
$FOD_PATH -ErrorAction stop | Out-Null
    }

    # Optional: Add Legacy Features
    For ( $index = 0; $index -lt $OC.count; $index++)
    {
        Write-Output "$(Get-TS): Adding $($OC[$index]) to main OS, index
$($IMAGE.ImageIndex)"
        DISM /Image:$MAIN_OS_MOUNT /Enable-Feature /FeatureName:$($OC[$index]) /All
| Out-Null
        if ($LastExitCode -ne 0)
        {
            throw "Error: Failed to add $($OC[$index]) to main OS, index
$($IMAGE.ImageIndex). Exit code: $LastExitCode"
        }
    }

    # Add latest cumulative update
    Write-Output "$(Get-TS): Adding package $LCU_PATH to main OS, index
$($IMAGE.ImageIndex)"
    Add-WindowsPackage -Path $MAIN_OS_MOUNT -PackagePath $LCU_PATH -ErrorAction stop
| Out-Null

    # Perform image cleanup. Some Optional Components might require the image to be
booted, and thus
    # image cleanup may fail. We'll catch and handle as a warning.
    Write-Output "$(Get-TS): Performing image cleanup on main OS, index
$($IMAGE.ImageIndex)"
    DISM /image:$MAIN_OS_MOUNT /cleanup-image /StartComponentCleanup | Out-Null
    if ($LastExitCode -ne 0)
    {
        if ($LastExitCode -eq -2146498554)
        {
            # We hit 0x800F0806 CBS_E_PENDING. We will ignore this with a warning
            # This is likely due to legacy components being added that require
online operations.
            Write-Warning "$(Get-TS): Failed to perform image cleanup on main OS,
index $($IMAGE.ImageIndex). Exit code: $LastExitCode. The operation cannot be
performed until pending servicing operations are completed. The image must be booted
to complete the pending servicing operation."
        }
        else
        {
            throw "Error: Failed to perform image cleanup on main OS, index
$($IMAGE.ImageIndex). Exit code: $LastExitCode"
        }
    }
```

```
    # Finally, we'll add .NET 3.5 and the .NET cumulative update
    Write-Output "$(Get-TS): Adding NetFX3~~~~ to main OS, index
$($IMAGE.ImageIndex)"
    Add-WindowsCapability -Name "NetFX3~~~~" -Path $MAIN_OS_MOUNT -Source $FOD_PATH
-ErrorAction stop | Out-Null

    # Add .NET Cumulative Update
    Write-Output "$(Get-TS): Adding package $DOTNET_CU_PATH to main OS, index
$($IMAGE.ImageIndex)"
    Add-WindowsPackage -Path $MAIN_OS_MOUNT -PackagePath $DOTNET_CU_PATH -
ErrorAction stop | Out-Null

    # Dismount
    Dismount-WindowsImage -Path $MAIN_OS_MOUNT -Save -ErrorAction stop | Out-Null

    # Export
    Write-Output "$(Get-TS): Exporting image to $WORKING_PATH\install2.wim"
    Export-WindowsImage -SourceImagePath $MEDIA_NEW_PATH"\sources\install.wim" -
SourceIndex $IMAGE.ImageIndex -DestinationImagePath $WORKING_PATH"\install2.wim" -
ErrorAction stop | Out-Null
}

Move-Item -Path $WORKING_PATH"\install2.wim" -Destination
$MEDIA_NEW_PATH"\sources\install.wim" -Force -ErrorAction stop | Out-Null
```

Update WinPE

This script is similar to the one that updates WinRE, but instead it mounts Boot.wim, applies the packages with the latest cumulative update last, and saves. It repeats this for all images inside of Boot.wim, typically two images. It starts by applying the servicing stack Dynamic Update. Since the script is customizing this media with Japanese, it installs the language pack from the WinPE folder on the language pack ISO. Additionally, it adds font support and text to speech (TTS) support. Since the script is adding a new language, it rebuilds lang.ini, used to identify languages installed in the image. For the second image, we save setup.exe and setuphost.exe for later use, to ensure these versions matches the \sources\setup.exe and \sources\setuphost.exe version from the installation media. If these binaries aren't identical, Windows Setup will fail during installation. We'll also save the serviced boot manager files for later use in the script. Finally, the script cleans and exports Boot.wim, and copies it back to the new media.

PowerShell

```
#
# update Windows Preinstallation Environment (WinPE)
#
```

```
# Get the list of images contained within WinPE
$WINPE_IMAGES = Get-WindowsImage -ImagePath $MEDIA_NEW_PATH"\sources\boot.wim"

Foreach ($IMAGE in $WINPE_IMAGES)
{

    # update WinPE
    Write-Output "$(Get-TS): Mounting WinPE, image index $($IMAGE.ImageIndex)"
    Mount-WindowsImage -ImagePath $MEDIA_NEW_PATH"\sources\boot.wim" -Index
$IMAGE.ImageIndex -Path $WINPE_MOUNT -ErrorAction stop | Out-Null

    # Add servicing stack update (Step 9 from the table)
    try
    {
        Write-Output "$(Get-TS): Adding package $LCU_PATH to WinPE, image index
$($IMAGE.ImageIndex)"
        Add-WindowsPackage -Path $WINPE_MOUNT -PackagePath $LCU_PATH | Out-Null
    }
    Catch
    {
        $theError = $_
        Write-Output "$(Get-TS): $theError"
        if ($theError.Exception -like "*0x8007007e*")
        {
            Write-Warning "$(Get-TS): Failed with error 0x8007007e. This failure is
a known issue with combined cumulative update, we can ignore."
        }
        else
        {
            throw
        }
    }

    # Install lp.cab cab
    Write-Output "$(Get-TS): Adding package $WINPE_OC_LP_PATH to WinPE, image index
$($IMAGE.ImageIndex)"
    Add-WindowsPackage -Path $WINPE_MOUNT -PackagePath $WINPE_OC_LP_PATH -
ErrorAction stop | Out-Null

    # Install language cabs for each optional package installed
    $WINPE_INSTALLED_OC = Get-WindowsPackage -Path $WINPE_MOUNT
    Foreach ($PACKAGE in $WINPE_INSTALLED_OC)
    {
        if ( ($PACKAGE.PackageState -eq "Installed") -and
($PACKAGE.PackageName.startsWith("WinPE-")) -and ($PACKAGE.ReleaseType -eq
"FeaturePack") )
        {
            $INDEX = $PACKAGE.PackageName.IndexOf("-Package")
            if ($INDEX -ge 0)
            {
                $OC_CAB = $PACKAGE.PackageName.Substring(0, $INDEX) + "_" + $LANG +
".cab"
                if ($WINPE_OC_LANG_CABS.Contains($OC_CAB))
                {
                    $OC_CAB_PATH = Join-Path $WINPE_OC_LANG_PATH $OC_CAB
```

```
                    Write-Output "$(Get-TS): Adding package $OC_CAB_PATH to WinPE,
image index $($IMAGE.ImageIndex)"
                    Add-WindowsPackage -Path $WINPE_MOUNT -PackagePath $OC_CAB_PATH
-ErrorAction stop | Out-Null
                }
            }
        }
    }

    # Add font support for the new language
    if ( (Test-Path -Path $WINPE_FONT_SUPPORT_PATH) )
    {
        Write-Output "$(Get-TS): Adding package $WINPE_FONT_SUPPORT_PATH to WinPE,
image index $($IMAGE.ImageIndex)"
        Add-WindowsPackage -Path $WINPE_MOUNT -PackagePath $WINPE_FONT_SUPPORT_PATH
-ErrorAction stop | Out-Null
    }

    # Add TTS support for the new language
    if (Test-Path -Path $WINPE_SPEECH_TTS_PATH)
    {
        if ( (Test-Path -Path $WINPE_SPEECH_TTS_LANG_PATH) )
        {
            Write-Output "$(Get-TS): Adding package $WINPE_SPEECH_TTS_PATH to WinPE,
image index $($IMAGE.ImageIndex)"
            Add-WindowsPackage -Path $WINPE_MOUNT -PackagePath
$WINPE_SPEECH_TTS_PATH -ErrorAction stop | Out-Null

            Write-Output "$(Get-TS): Adding package $WINPE_SPEECH_TTS_LANG_PATH to
WinPE, image index $($IMAGE.ImageIndex)"
            Add-WindowsPackage -Path $WINPE_MOUNT -PackagePath
$WINPE_SPEECH_TTS_LANG_PATH -ErrorAction stop | Out-Null
        }
    }

    # Generates a new Lang.ini file which is used to define the language packs
inside the image
    if ( (Test-Path -Path $WINPE_MOUNT"\sources\lang.ini") )
    {
        Write-Output "$(Get-TS): Updating lang.ini"
        DISM /image:$WINPE_MOUNT /Gen-LangINI /distribution:$WINPE_MOUNT | Out-Null
        if ($LastExitCode -ne 0)
        {
            throw "Error: Failed to update lang.ini. Exit code: $LastExitCode"
        }
    }

    # Add latest cumulative update
    Write-Output "$(Get-TS): Adding package $LCU_PATH to WinPE, image index
$($IMAGE.ImageIndex)"
    Add-WindowsPackage -Path $WINPE_MOUNT -PackagePath $LCU_PATH -ErrorAction stop |
Out-Null

    # Perform image cleanup
```

```
    Write-Output "$(Get-TS): Performing image cleanup on WinPE, image index
$($IMAGE.ImageIndex)"
    DISM /image:$WINPE_MOUNT /cleanup-image /StartComponentCleanup /ResetBase /Defer
| Out-Null
    if ($LastExitCode -ne 0)
    {
        throw "Error: Failed to perform image cleanup on WinPE, image index
$($IMAGE.ImageIndex). Exit code: $LastExitCode"
    }

    if ($IMAGE.ImageIndex -eq "2")
    {
        # Save setup.exe for later use. This will address possible binary mismatch
with the version in the main OS \sources folder
        Copy-Item -Path $WINPE_MOUNT"\sources\setup.exe" -Destination
$WORKING_PATH"\setup.exe" -Force -ErrorAction stop | Out-Null

        # Save setuphost.exe for later use. This will address possible binary
mismatch with the version in the main OS \sources folder
        # This is only required starting with Windows 11 version 24H2
        $TEMP = Get-WindowsImage -ImagePath $MEDIA_NEW_PATH"\sources\boot.wim" -
Index $IMAGE.ImageIndex
        if ([System.Version]$TEMP.Version -ge [System.Version]"10.0.26100")
        {
            Copy-Item -Path $WINPE_MOUNT"\sources\setuphost.exe" -Destination
$WORKING_PATH"\setuphost.exe" -Force -ErrorAction stop | Out-Null
        }
        else
        {
            Write-Output "$(Get-TS): Skipping copy of setuphost.exe; image version
$($TEMP.Version)"
        }

        # Save serviced boot manager files later copy to the root media.
        Copy-Item -Path $WINPE_MOUNT"\Windows\boot\efi\bootmgfw.efi" -Destination
$WORKING_PATH"\bootmgfw.efi" -Force -ErrorAction stop | Out-Null
        Copy-Item -Path $WINPE_MOUNT"\Windows\boot\efi\bootmgr.efi" -Destination
$WORKING_PATH"\bootmgr.efi" -Force -ErrorAction stop | Out-Null
    }

    # Dismount
    Dismount-WindowsImage -Path $WINPE_MOUNT -Save -ErrorAction stop | Out-Null

    #Export WinPE
    Write-Output "$(Get-TS): Exporting image to $WORKING_PATH\boot2.wim"
    Export-WindowsImage -SourceImagePath $MEDIA_NEW_PATH"\sources\boot.wim" -
SourceIndex $IMAGE.ImageIndex -DestinationImagePath $WORKING_PATH"\boot2.wim" -
ErrorAction stop | Out-Null
}

Move-Item -Path $WORKING_PATH"\boot2.wim" -Destination
$MEDIA_NEW_PATH"\sources\boot.wim" -Force -ErrorAction stop | Out-Null
```

Update remaining media files

This part of the script updates the Setup files. It simply copies the individual files in the Setup Dynamic Update package to the new media. This step brings in updated Setup files as needed, along with the latest compatibility database, and replacement component manifests. This script also does a final replacement of setup.exe, setuphost.exe, and boot manager files using the previously saved versions from WinPE.

PowerShell

```
#
# update remaining files on media
#

# Add Setup DU by copy the files from the package into the newMedia
Write-Output "$(Get-TS): Adding package $SETUP_DU_PATH"
cmd.exe /c $env:SystemRoot\System32\expand.exe $SETUP_DU_PATH -F:*
$MEDIA_NEW_PATH"\sources" | Out-Null
if ($LastExitCode -ne 0)
{
    throw "Error: Failed to expand $SETUP_DU_PATH. Exit code: $LastExitCode"
}

# Copy setup.exe from boot.wim, saved earlier.
Write-Output "$(Get-TS): Copying $WORKING_PATH\setup.exe to
$MEDIA_NEW_PATH\sources\setup.exe"
Copy-Item -Path $WORKING_PATH"\setup.exe" -Destination
$MEDIA_NEW_PATH"\sources\setup.exe" -Force -ErrorAction stop | Out-Null

# Copy setuphost.exe from boot.wim, saved earlier.
if (Test-Path -Path $WORKING_PATH"\setuphost.exe")
{
    Write-Output "$(Get-TS): Copying $WORKING_PATH\setuphost.exe to
$MEDIA_NEW_PATH\sources\setuphost.exe"
    Copy-Item -Path $WORKING_PATH"\setuphost.exe" -Destination
$MEDIA_NEW_PATH"\sources\setuphost.exe" -Force -ErrorAction stop | Out-Null
}

# Copy bootmgr files from boot.wim, saved earlier.
$MEDIA_NEW_FILES = Get-ChildItem $MEDIA_NEW_PATH -Force -Recurse -Filter b*.efi

Foreach ($File in $MEDIA_NEW_FILES)
{
    if (($File.Name -ieq "bootmgfw.efi") -or ($File.Name -ieq "bootx64.efi") -or
($File.Name -ieq "bootia32.efi") -or ($File.Name -ieq "bootaa64.efi"))
    {
        Write-Output "$(Get-TS): Copying $WORKING_PATH\bootmgfw.efi to
$($File.FullName)"
        Copy-Item -Path $WORKING_PATH"\bootmgfw.efi" -Destination $File.FullName -
Force -ErrorAction stop | Out-Null
    }
    elseif ($File.Name -ieq "bootmgr.efi")
```

```
    {
        Write-Output "$(Get-TS): Copying $WORKING_PATH\bootmgr.efi to
$($File.FullName)"
        Copy-Item -Path $WORKING_PATH"\bootmgr.efi" -Destination $File.FullName -
Force -ErrorAction stop | Out-Null
    }
}
```

Finish up

As a last step, the script removes the working folder of temporary files, and unmounts our language pack and Features on Demand ISOs.

PowerShell

```
#
# Perform final cleanup
#

# Remove our working folder
Remove-Item -Path $WORKING_PATH -Recurse -Force -ErrorAction stop | Out-Null

# Dismount ISO images
Write-Output "$(Get-TS): Dismounting ISO images"
Dismount-DiskImage -ImagePath $FOD_ISO_PATH -ErrorAction stop | Out-Null

Write-Output "$(Get-TS): Media refresh completed!"
```

Last updated on 08/11/2025

Migrating and acquiring optional Windows content during updates

Article • 04/22/2024 • Applies to: ✅ Windows 11, ✅ Windows 10

This article provides some background on the problem of keeping language resources and Features on Demand during operating system updates and offers guidance to help you move forward in the short term and prepare for the long term.

When you update the operating system, it's critical to keep language resources and Features on Demand (FODs). Many commercial organizations use Configuration Manager or other management tools to distribute and orchestrate Windows client setup using a local Windows image or WIM file (a *media-based* or *task-sequence-based* update). Others do in-place updates using an approved Windows client feature update by using Windows Server Update Services (WSUS), Configuration Manager, or equivalent tools (a *servicing-based* update).

Neither approach contains the full set of Windows optional features that a user's device might need, so those features aren't migrated to the new operating system. In the past, those features weren't available in Configuration Manager nor WSUS for on-premises acquisition after a feature update.

What is optional content?

Optional content includes the following items:

- General Features on Demand also referred to as FODs (for example, Windows Mixed Reality)
- Language-based and regional FODs (for example, Language.Basic~~~ja-jp~0.0.1.0)
- Local Experience Packs
- Language packs

Optional content isn't included by default in the Windows image file that is part of the operating system media available in the Volume Licensing Service Center (VLSC). Instead, it's released as an additional ISO file on VLSC. Shipping these features out of the operating system media and shipping them separately reduces the disk footprint of Windows. This approach provides more space for user's data. It also reduces the time needed to service the operating system, whether installing a monthly quality update or upgrading to a newer version. A smaller default Windows image also means less data to transmit over the network.

Why is acquiring optional content challenging?

The challenges surrounding optional content typically fall into two groups:

Incomplete operating system updates

The first challenge is related to content migration during a feature update. When Windows Setup performs an in-place update, the new operating system is written to the user's disk alongside the old version in a temporary folder, where a second clean operating system is installed and prepared for the user to *move into*. When operation happens, Windows Setup enumerates optional content installed already in the current version and plans to install the new version of this content in the new operating system.

Windows Setup needs access to the optional content. Since optional content isn't in the Windows image by default, Windows Setup must look elsewhere to get the Windows packages, stage them, and then install them in the new operating system. When the content can't be found, the result is an update that is missing features on the device, a frustrated end user, and likely a help desk call. This pain point is sometimes referred to as *failure to migrate optional content during update*. For media-based updates, Windows will automatically try again once the new operating system boots. We call this *latent acquisition*.

User-initiated feature acquisition failure

The second challenge involves a failure to acquire features when a user requests them. Imagine a user running a device with a new version of Windows client, either by using a clean installation or an in-place update. The user visits **Settings**, and attempts to install a second language, more language experience features, or other optional content. Again, since these features aren't in the operating system, the packages need to be acquired. For a typical user with internet access, Windows acquires the features from a nearby Microsoft content delivery network, and everything works as designed. For commercial users, some might not have internet access or have policies to prevent acquisition over the internet. In these situations, Windows must acquire the content from an alternative location. When the content can't be found, users are frustrated, and another help desk call could result. This pain point is sometimes referred to as *failure to acquire optional content*.

Options for acquiring optional content

Most commercial organizations understand the pain points outlined above, and discussions typically start with them asking what plans are available to address these challenges. The following table includes multiple options for consideration, depending on how you're currently deploying Windows client. The following definitions are used in the table headings:

- **Migration**: Supports optional content migration during an update.
- **Acquisition**: Supports optional content acquisition (that is, initiated by the user).
- **Media**: Applicable with media-based deployments.
- **Servicing**: Applicable with servicing-based deployments.

Expand table

Method	Migration	Acquisition	Media	Servicing
Option 1: Use Windows Update	Yes	Yes	No	Yes
Option 2: Use WSUS with UUP Integration	Yes	Yes	No	Yes
Option 3: Enable Dynamic Update	Yes	No	Yes	Yes
Option 4: Customize the Windows image before deployment	Yes	No	Yes	No
Option 5: Install language features during deployment	Partial	No	Yes	No
Option 6: Install optional content after deployment	Yes	No	Yes	Yes
Option 7: Configure alternative source for Features on Demand	No	Partial	Yes	Yes

Option 1: Use Windows Update

Windows Update client policies solve the optional content problem. Optional content is published and available for acquisition by Windows Setup from a nearby Microsoft content delivery network and acquired using the Unified Update Platform. Optional content migration and acquisition scenarios just work when the device is connected to an update service that uses the Unified Update Platform, such as Windows Update or Windows Update client policies. If for some reason a language pack fails to install during the update, the update will automatically roll back.

The Unified Update Platform is an improvement in the underlying Windows update technology that results in smaller download sizes and a more efficient protocol for checking for updates, acquiring and installing the packages needed, and getting current

in one update step. The technology is *unified* because it brings together the update stack for Windows client, Windows Server, and other products, such as HoloLens.

Consider moving to Windows Update client policies. Not only will the optional content scenario work seamlessly (as it does for consumer devices today), but you also get the full benefits of smaller download sizes. Further, devices are immune to the challenge of upgrading Windows when the operating system installation language is inadvertently changed to a new language. Otherwise, any future media-based feature updates can fail when the installation media has a different installation language. For more information about this issue, see Upgrading Windows 10 devices with installation media different than the original OS install language.

Option 2: Use WSUS with UUP Integration

Starting in March 2023, UUP has been integrated with WSUS and Configuration Manager to bring the same optional content and acquisition benefits of Windows Update to on-premises management solutions. For example:

- FODs and languages will automatically migrate for devices that perform an in-place update using an approved Windows 11, version 22H2 client feature update from WSUS. Similarly, updates such as the combined cumulative update, Setup updates, and Safe OS updates are included and current based on the month that the feature update was approved.
- Devices that upgrade using a local Windows image but use WSUS or Configuration Manager for approving the combined cumulative update will benefit by having support for optional content acquisition in the updated Windows OS, as well as OS self-healing.

The content required to enable this will be acquired via WSUS or Configuration Manager, without client endpoints requiring internet connectivity. To enable this improvement, once per major Windows release, a significant download to the WSUS content directory or the distribution point is required. This includes packages to support FOD and language acquisition, along with packages to enable OS self-healing due to corruption. For more information, see Plan your WSUS deployment.

Option 3: Enable Dynamic Update

If you're not ready to move to Windows Update, another option is to enable Dynamic Update during a feature update. As soon as a Windows feature update starts, whether via a media-based update or a WSUS-based feature update, Dynamic Update is one of the first steps invoked. Windows Setup connects to an internet-facing URL hosted by

Microsoft to fetch Dynamic Update content, and then applies those updates to the operating system installation media. The content acquired includes the following:

- **Setup updates**: Fixes to Setup.exe binaries or any files that Setup uses for feature updates.
- **Safe OS updates**: Fixes for the *safe OS* that are used to update Windows recovery environment (WinRE).
- **Servicing stack updates**: Fixes that are necessary to address the Windows servicing stack issue and thus required to complete the feature update.
- **Latest cumulative update**: Installs the latest cumulative quality update.
- **Driver updates**: Latest version of applicable drivers that have already been published by manufacturers into Windows Update and meant specifically for Dynamic Update.

In addition to these updates for the new operating system, Dynamic Update acquires optional content during the update process to ensure that the device has this content present when the update completes. So, although the device isn't connected to Windows Update, it fetches content from a nearby Microsoft content download network (CDN). This approach addresses the first pain point with optional content, but not user-initiated acquisition. By default, Dynamic Update is enabled by Windows Setup. You can enable or disable Dynamic Update by using the /DynamicUpdate option in Windows Setup. If you use the servicing-based approach, you can set this value with `setupconfig.ini`. See Windows Setup Automation Overview for details.

Dynamic Update can be configured with additional options. For example, you might want to have the benefits of optional content migration without automatically acquiring the latest quality update. You can do that with the /DynamicUpdate NoLCU option of Windows Setup. Afterward, you would separately follow your existing process for testing and approving monthly updates. The downside of this approach is the device reboots again for the latest cumulative update since it wasn't available during the feature update.

One further consideration when using Dynamic Update is the effect on your network. One of the top blockers for this approach is the concern that each device will separately fetch this content from Microsoft. Setup downloads Dynamic Update content using Delivery Optimization when available. For devices that aren't connected to the internet, a subset of the Dynamic Update content is available by using WSUS and the Microsoft catalog.

Option 4: Customize the Windows Image before deployment

For many organizations, the deployment workflow involves a Configuration Manager task sequence that performs a media-based update. Some customers either don't have internet connectivity, or the connectivity is poor and so they can't enable Dynamic Update. In these cases, we recommend installing optional content prior to deployment. This activity is sometimes referred to as customizing the installation media.

You can customize the Windows image in these ways:

- Applying a cumulative update
- Applying updates to the servicing stack
- Applying updates to `Setup.exe` binaries or other files that setup uses for feature updates
- Applying updates for the *safe operating system* (SafeOS) that's used for the Windows recovery environment
- Adding or removing languages
- Adding or removing Features on Demand

The benefit of this option is that the Windows image can include those additional languages, language experience features, and other Features on Demand through one-time updates to the image. Then you can use them in an existing task sequence or custom deployment where `Setup.exe` is involved. The downside of this approach is that it requires some preparation of the image in advance, including scripting with DISM to install the additional packages. It also means the image is the same for all devices that consume it and might contain more features than some users need. For more information on customizing your media, see Updating Windows 10 media with Dynamic Update packages. Also like Dynamic Update, you still have a solution for migration of optional content, but not supporting user-initiated optional content acquisition. Also, there's a variation of this option in which media is updated *on the device* just before installation. This option allows for device-specific image customization based on what's currently installed.

Option 5: Install language features during deployment

A partial solution to address the first pain point of failing to migrate optional content during upgrade is to inject a subset of optional content during the upgrade process. This approach uses the Windows Setup option /InstallLangPacks to add Language Packs and language capabilities such as text-to-speech recognition from a folder that contains the packages. This approach lets an IT pro take a subset of optional content and stage them within their network. If you use the servicing-based approach, you can configure InstallLangPacks using `setupconfig.ini`. For more information, see Windows Setup Automation Overview.

When Setup runs, it injects these packages into the new operating system during installation. It can be an alternative to enabling Dynamic Update or customizing the operating system image before deployment. You must take care with this approach, because the packages can't be renamed. Further, the content is coming from two separate release media ISOs. The key is to copy both the FOD packages and the FOD metadata .cab from the FOD ISO into the folder, and the architecture-specific Language Pack .cab files from the LPLIP ISO. We treat InstallLangPacks failures as fatal, and roll back the entire upgrade. The idea is to not leave the user in a bad state since media-based upgrades don't migrate FOD and languages (unless Dynamic Update is enabled).

This approach has some interesting benefits. The original Windows image doesn't need to be modified, possibly saving time and scripting.

Option 6: Install optional content after deployment

This option is like Option 4 in that you customize the operating system image with more optional content after it's deployed. IT pros can extend the behavior of Windows Setup by running their own custom action scripts during and after a feature update. See Run custom actions during feature update for details. With this approach, you can create a device-specific migration of optional content by capturing the optional content that's installed in the operating system, and then saving this list to install the same optional content in the new operating system. Like Option 5, you would internally host a network share that contains the source of the optional content packages. Then, during the execution of Setup on the device, capture the list of installed optional content from the source operating system and save. Later, after Setup completes, you use the list to install the optional content, which leaves the user's device without loss of functionality.

Option 7: Configure an alternative source for optional content

Several of the options address ways to address optional content migration issues during an in-place update. To address the second pain point of easily acquiring optional content in the user-initiated case, you can configure each device by using the Specify settings for optional component installation and component repair Group Policy. This policy setting specifies the network locations that will be used for the repair of operating system corruption and for enabling optional features that have had their payload files removed. This approach has the disadvantage of more content to be hosted within your network (in addition to the operating system image you might be still deploying to some clients) but has the advantage of acquiring content within your network. Some reminders about this policy:

- The file path to the alternate source must be a fully qualified path; multiple locations can be separated by a semicolon.
- This setting doesn't support installing language packs from an alternate source file path, only Features on Demand. If the policy is configured to acquire content from Windows Update, language packs will be acquired.
- If this setting isn't configured or disabled, files are downloaded from the default Windows Update location, for example Windows Update client policies or WSUS.

For more information, see Configure a Windows Repair Source and How to make Features on Demand and language packs available when you're using WSUS or Configuration Manager.

More resources

For more information about the Unified Update Platform and the approaches outlined in this article, see the following resources:

- Plan your WSUS deployment
- /InstallLangPacks
- /DynamicUpdate
- Configure a Windows Repair Source
- Run custom actions during feature update
- Unified Update Platform
- Updating Windows installation media with Dynamic Update packages
- Windows Setup Automation Overview

Sample scripts

Options 4 and 6 involve the most scripting. Sample scripts for Option 4 already exist, so let's look at sample scripts for Option 6: Install Optional Content after Deployment.

Creating an optional content repository

To get started, we build a repository of optional content and host on a network share. This content is a subset of content from the FOD and language pack ISOs that ship with each release. We configure this repository or repo with only those FODs our organization needs, using DISM /Export. For example, a superset based on taking inventory of optional features installed on existing devices. In this case, we exclude the Windows Mixed Reality feature. In addition, we copy all language packs to the root of the repository.

PowerShell

```
# Declare media for FOD and LPs
$LP_ISO_PATH = "C:\_IMAGE\2004_ISO\CLIENTLANGPACKDVD_OEM_MULTI.iso"
$FOD_ISO_PATH = "C:\_IMAGE\2004_ISO\FOD-PACKAGES_OEM_PT1_amd64fre_MULTI.iso"

# Declare folders
$WORKING_PATH = "C:\_IMAGE\BuildRepo"
$MEDIA_PATH = "C:\_IMAGE\2004_SETUP"

$MAIN_OS_MOUNT = $WORKING_PATH + "\MainOSMount"
$REPO_PATH = $WORKING_PATH + "\Repo"

# Create folders for mounting image optional content repository
if (Test-Path $MAIN_OS_MOUNT) {
    Remove-Item -Path $MAIN_OS_MOUNT -Force -Recurse -ErrorAction stop| Out-
Null
}

if (Test-Path $REPO_PATH) {
    Remove-Item -Path $REPO_PATH -Force -Recurse -ErrorAction stop| Out-Null
}

New-Item -ItemType Directory -Force -Path $MAIN_OS_MOUNT -ErrorAction stop|
Out-Null
New-Item -ItemType Directory -Force -Path $REPO_PATH -ErrorAction stop| Out-
Null

# Mount the main OS, I'll use this throughout the script
Write-Host "Mounting main OS"
Mount-WindowsImage -ImagePath $MEDIA_PATH"\sources\install.wim" -Index 1 -
Path $MAIN_OS_MOUNT -ErrorAction stop| Out-Null

# Mount the LP ISO
Write-Host "Mounting LP ISO"
$LP_ISO_DRIVE_LETTER = (Mount-DiskImage -ImagePath $LP_ISO_PATH -ErrorAction
stop | Get-Volume).DriveLetter

# Declare language related cabs
$OS_LP_PATH = $LP_ISO_DRIVE_LETTER + ":\x64\langpacks\" + "*.cab"

# Mount the FOD ISO
Write-Host "Mounting FOD ISO"
$FOD_ISO_DRIVE_LETTER = (Mount-DiskImage -ImagePath $FOD_ISO_PATH -
ErrorAction stop | Get-Volume).DriveLetter
$FOD_PATH = $FOD_ISO_DRIVE_LETTER + ":\"

# Export the FODs from the ISO that we are interested in
Write-Host "Exporting FODs to Repo"
DISM /image:$MAIN_OS_MOUNT /export-source /source:$FOD_PATH
/target:$REPO_PATH `
    /capabilityname:Accessibility.Braille~~~~0.0.1.0 `
    /capabilityname:App.StepsRecorder~~~~0.0.1.0 `
    /capabilityname:App.WirelessDisplay.Connect~~~~0.0.1.0 `
```

```
    /capabilityname:Browser.InternetExplorer~~~~0.0.11.0 `
    /capabilityname:DirectX.Configuration.Database~~~~0.0.1.0 `
    /capabilityname:Language.Basic~~~af-za~0.0.1.0 `
    /capabilityname:Language.Basic~~~ar-sa~0.0.1.0 `
    /capabilityname:Language.Basic~~~as-in~0.0.1.0 `
    /capabilityname:Language.Basic~~~az-latn-az~0.0.1.0 `
    /capabilityname:Language.Basic~~~ba-ru~0.0.1.0 `
    /capabilityname:Language.Basic~~~be-by~0.0.1.0 `
    /capabilityname:Language.Basic~~~bg-bg~0.0.1.0 `
    /capabilityname:Language.Basic~~~bn-bd~0.0.1.0 `
    /capabilityname:Language.Basic~~~bn-in~0.0.1.0 `
    /capabilityname:Language.Basic~~~bs-latn-ba~0.0.1.0 `
    /capabilityname:Language.Basic~~~ca-es~0.0.1.0 `
    /capabilityname:Language.Basic~~~cs-cz~0.0.1.0 `
    /capabilityname:Language.Basic~~~cy-gb~0.0.1.0 `
    /capabilityname:Language.Basic~~~da-dk~0.0.1.0 `
    /capabilityname:Language.Basic~~~de-ch~0.0.1.0 `
    /capabilityname:Language.Basic~~~de-de~0.0.1.0 `
    /capabilityname:Language.Basic~~~el-gr~0.0.1.0 `
    /capabilityname:Language.Basic~~~en-au~0.0.1.0 `
    /capabilityname:Language.Basic~~~en-ca~0.0.1.0 `
    /capabilityname:Language.Basic~~~en-gb~0.0.1.0 `
    /capabilityname:Language.Basic~~~en-in~0.0.1.0 `
    /capabilityname:Language.Basic~~~en-us~0.0.1.0 `
    /capabilityname:Language.Basic~~~es-es~0.0.1.0 `
    /capabilityname:Language.Basic~~~es-mx~0.0.1.0 `
    /capabilityname:Language.Basic~~~es-us~0.0.1.0 `
    /capabilityname:Language.Basic~~~et-ee~0.0.1.0 `
    /capabilityname:Language.Basic~~~eu-es~0.0.1.0 `
    /capabilityname:Language.Basic~~~fa-ir~0.0.1.0 `
    /capabilityname:Language.Basic~~~fi-fi~0.0.1.0 `
    /capabilityname:Language.Basic~~~fil-ph~0.0.1.0 `
    /capabilityname:Language.Basic~~~fr-be~0.0.1.0 `
    /capabilityname:Language.Basic~~~fr-ca~0.0.1.0 `
    /capabilityname:Language.Basic~~~fr-ch~0.0.1.0 `
    /capabilityname:Language.Basic~~~fr-fr~0.0.1.0 `
    /capabilityname:Language.Basic~~~ga-ie~0.0.1.0 `
    /capabilityname:Language.Basic~~~gd-gb~0.0.1.0 `
    /capabilityname:Language.Basic~~~gl-es~0.0.1.0 `
    /capabilityname:Language.Basic~~~gu-in~0.0.1.0 `
    /capabilityname:Language.Basic~~~ha-latn-ng~0.0.1.0 `
    /capabilityname:Language.Basic~~~haw-us~0.0.1.0 `
    /capabilityname:Language.Basic~~~he-il~0.0.1.0 `
    /capabilityname:Language.Basic~~~hi-in~0.0.1.0 `
    /capabilityname:Language.Basic~~~hr-hr~0.0.1.0 `
    /capabilityname:Language.Basic~~~hu-hu~0.0.1.0 `
    /capabilityname:Language.Basic~~~hy-am~0.0.1.0 `
    /capabilityname:Language.Basic~~~id-id~0.0.1.0 `
    /capabilityname:Language.Basic~~~ig-ng~0.0.1.0 `
    /capabilityname:Language.Basic~~~is-is~0.0.1.0 `
    /capabilityname:Language.Basic~~~it-it~0.0.1.0 `
    /capabilityname:Language.Basic~~~ja-jp~0.0.1.0 `
    /capabilityname:Language.Basic~~~ka-ge~0.0.1.0 `
    /capabilityname:Language.Basic~~~kk-kz~0.0.1.0 `
    /capabilityname:Language.Basic~~~kl-gl~0.0.1.0 `
```

```
    /capabilityname:Language.Basic~~~kn-in~0.0.1.0 `
    /capabilityname:Language.Basic~~~kok-deva-in~0.0.1.0 `
    /capabilityname:Language.Basic~~~ko-kr~0.0.1.0 `
    /capabilityname:Language.Basic~~~ky-kg~0.0.1.0 `
    /capabilityname:Language.Basic~~~lb-lu~0.0.1.0 `
    /capabilityname:Language.Basic~~~lt-lt~0.0.1.0 `
    /capabilityname:Language.Basic~~~lv-lv~0.0.1.0 `
    /capabilityname:Language.Basic~~~mi-nz~0.0.1.0 `
    /capabilityname:Language.Basic~~~mk-mk~0.0.1.0 `
    /capabilityname:Language.Basic~~~ml-in~0.0.1.0 `
    /capabilityname:Language.Basic~~~mn-mn~0.0.1.0 `
    /capabilityname:Language.Basic~~~mr-in~0.0.1.0 `
    /capabilityname:Language.Basic~~~ms-bn~0.0.1.0 `
    /capabilityname:Language.Basic~~~ms-my~0.0.1.0 `
    /capabilityname:Language.Basic~~~mt-mt~0.0.1.0 `
    /capabilityname:Language.Basic~~~nb-no~0.0.1.0 `
    /capabilityname:Language.Basic~~~ne-np~0.0.1.0 `
    /capabilityname:Language.Basic~~~nl-nl~0.0.1.0 `
    /capabilityname:Language.Basic~~~nn-no~0.0.1.0 `
    /capabilityname:Language.Basic~~~nso-za~0.0.1.0 `
    /capabilityname:Language.Basic~~~or-in~0.0.1.0 `
    /capabilityname:Language.Basic~~~pa-in~0.0.1.0 `
    /capabilityname:Language.Basic~~~pl-pl~0.0.1.0 `
    /capabilityname:Language.Basic~~~ps-af~0.0.1.0 `
    /capabilityname:Language.Basic~~~pt-br~0.0.1.0 `
    /capabilityname:Language.Basic~~~pt-pt~0.0.1.0 `
    /capabilityname:Language.Basic~~~rm-ch~0.0.1.0 `
    /capabilityname:Language.Basic~~~ro-ro~0.0.1.0 `
    /capabilityname:Language.Basic~~~ru-ru~0.0.1.0 `
    /capabilityname:Language.Basic~~~rw-rw~0.0.1.0 `
    /capabilityname:Language.Basic~~~sah-ru~0.0.1.0 `
    /capabilityname:Language.Basic~~~si-lk~0.0.1.0 `
    /capabilityname:Language.Basic~~~sk-sk~0.0.1.0 `
    /capabilityname:Language.Basic~~~sl-si~0.0.1.0 `
    /capabilityname:Language.Basic~~~sq-al~0.0.1.0 `
    /capabilityname:Language.Basic~~~sr-cyrl-rs~0.0.1.0 `
    /capabilityname:Language.Basic~~~sr-latn-rs~0.0.1.0 `
    /capabilityname:Language.Basic~~~sv-se~0.0.1.0 `
    /capabilityname:Language.Basic~~~sw-ke~0.0.1.0 `
    /capabilityname:Language.Basic~~~ta-in~0.0.1.0 `
    /capabilityname:Language.Basic~~~te-in~0.0.1.0 `
    /capabilityname:Language.Basic~~~tg-cyrl-tj~0.0.1.0 `
    /capabilityname:Language.Basic~~~th-th~0.0.1.0 `
    /capabilityname:Language.Basic~~~tk-tm~0.0.1.0 `
    /capabilityname:Language.Basic~~~tn-za~0.0.1.0 `
    /capabilityname:Language.Basic~~~tr-tr~0.0.1.0 `
    /capabilityname:Language.Basic~~~tt-ru~0.0.1.0 `
    /capabilityname:Language.Basic~~~ug-cn~0.0.1.0 `
    /capabilityname:Language.Basic~~~uk-ua~0.0.1.0 `
    /capabilityname:Language.Basic~~~ur-pk~0.0.1.0 `
    /capabilityname:Language.Basic~~~uz-latn-uz~0.0.1.0 `
    /capabilityname:Language.Basic~~~vi-vn~0.0.1.0 `
    /capabilityname:Language.Basic~~~wo-sn~0.0.1.0 `
    /capabilityname:Language.Basic~~~xh-za~0.0.1.0 `
    /capabilityname:Language.Basic~~~yo-ng~0.0.1.0 `
```

```
    /capabilityname:Language.Basic~~~zh-cn~0.0.1.0 `
    /capabilityname:Language.Basic~~~zh-hk~0.0.1.0 `
    /capabilityname:Language.Basic~~~zh-tw~0.0.1.0 `
    /capabilityname:Language.Basic~~~zu-za~0.0.1.0 `
    /capabilityname:Language.Fonts.Arab~~~und-Arab~0.0.1.0 `
    /capabilityname:Language.Fonts.Beng~~~und-Beng~0.0.1.0 `
    /capabilityname:Language.Fonts.Cans~~~und-Cans~0.0.1.0 `
    /capabilityname:Language.Fonts.Cher~~~und-Cher~0.0.1.0 `
    /capabilityname:Language.Fonts.Deva~~~und-Deva~0.0.1.0 `
    /capabilityname:Language.Fonts.Ethi~~~und-Ethi~0.0.1.0 `
    /capabilityname:Language.Fonts.Gujr~~~und-Gujr~0.0.1.0 `
    /capabilityname:Language.Fonts.Guru~~~und-Guru~0.0.1.0 `
    /capabilityname:Language.Fonts.Hans~~~und-Hans~0.0.1.0 `
    /capabilityname:Language.Fonts.Hant~~~und-Hant~0.0.1.0 `
    /capabilityname:Language.Fonts.Hebr~~~und-Hebr~0.0.1.0 `
    /capabilityname:Language.Fonts.Jpan~~~und-Jpan~0.0.1.0 `
    /capabilityname:Language.Fonts.Khmr~~~und-Khmr~0.0.1.0 `
    /capabilityname:Language.Fonts.Knda~~~und-Knda~0.0.1.0 `
    /capabilityname:Language.Fonts.Kore~~~und-Kore~0.0.1.0 `
    /capabilityname:Language.Fonts.Laoo~~~und-Laoo~0.0.1.0 `
    /capabilityname:Language.Fonts.Mlym~~~und-Mlym~0.0.1.0 `
    /capabilityname:Language.Fonts.Orya~~~und-Orya~0.0.1.0 `
    /capabilityname:Language.Fonts.PanEuropeanSupplementalFonts~~~0.0.1.0 `
    /capabilityname:Language.Fonts.Sinh~~~und-Sinh~0.0.1.0 `
    /capabilityname:Language.Fonts.Syrc~~~und-Syrc~0.0.1.0 `
    /capabilityname:Language.Fonts.Taml~~~und-Taml~0.0.1.0 `
    /capabilityname:Language.Fonts.Telu~~~und-Telu~0.0.1.0 `
    /capabilityname:Language.Fonts.Thai~~~und-Thai~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~af-za~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~bs-latn-ba~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~ca-es~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~cs-cz~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~cy-gb~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~da-dk~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~de-de~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~el-gr~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~en-gb~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~en-us~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~es-es~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~es-mx~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~eu-es~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~fi-fi~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~fr-fr~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~ga-ie~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~gd-gb~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~gl-es~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~hi-in~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~hr-hr~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~id-id~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~it-it~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~ja-jp~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~ko-kr~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~lb-lu~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~mi-nz~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~ms-bn~0.0.1.0 `
```

```
    /capabilityname:Language.Handwriting~~~ms-my~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~nb-no~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~nl-nl~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~nn-no~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~nso-za~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~pl-pl~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~pt-br~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~pt-pt~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~rm-ch~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~ro-ro~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~ru-ru~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~rw-rw~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~sk-sk~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~sl-si~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~sq-al~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~sr-cyrl-rs~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~sr-latn-rs~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~sv-se~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~sw-ke~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~tn-za~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~tr-tr~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~wo-sn~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~xh-za~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~zh-cn~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~zh-hk~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~zh-tw~0.0.1.0 `
    /capabilityname:Language.Handwriting~~~zu-za~0.0.1.0 `
    /capabilityname:Language.LocaleData~~~zh-tw~0.0.1.0 `
    /capabilityname:Language.OCR~~~ar-sa~0.0.1.0 `
    /capabilityname:Language.OCR~~~bg-bg~0.0.1.0 `
    /capabilityname:Language.OCR~~~bs-latn-ba~0.0.1.0 `
    /capabilityname:Language.OCR~~~cs-cz~0.0.1.0 `
    /capabilityname:Language.OCR~~~da-dk~0.0.1.0 `
    /capabilityname:Language.OCR~~~de-de~0.0.1.0 `
    /capabilityname:Language.OCR~~~el-gr~0.0.1.0 `
    /capabilityname:Language.OCR~~~en-gb~0.0.1.0 `
    /capabilityname:Language.OCR~~~en-us~0.0.1.0 `
    /capabilityname:Language.OCR~~~es-es~0.0.1.0 `
    /capabilityname:Language.OCR~~~es-mx~0.0.1.0 `
    /capabilityname:Language.OCR~~~fi-fi~0.0.1.0 `
    /capabilityname:Language.OCR~~~fr-ca~0.0.1.0 `
    /capabilityname:Language.OCR~~~fr-fr~0.0.1.0 `
    /capabilityname:Language.OCR~~~hr-hr~0.0.1.0 `
    /capabilityname:Language.OCR~~~hu-hu~0.0.1.0 `
    /capabilityname:Language.OCR~~~it-it~0.0.1.0 `
    /capabilityname:Language.OCR~~~ja-jp~0.0.1.0 `
    /capabilityname:Language.OCR~~~ko-kr~0.0.1.0 `
    /capabilityname:Language.OCR~~~nb-no~0.0.1.0 `
    /capabilityname:Language.OCR~~~nl-nl~0.0.1.0 `
    /capabilityname:Language.OCR~~~pl-pl~0.0.1.0 `
    /capabilityname:Language.OCR~~~pt-br~0.0.1.0 `
    /capabilityname:Language.OCR~~~pt-pt~0.0.1.0 `
    /capabilityname:Language.OCR~~~ro-ro~0.0.1.0 `
    /capabilityname:Language.OCR~~~ru-ru~0.0.1.0 `
    /capabilityname:Language.OCR~~~sk-sk~0.0.1.0 `
```

```
    /capabilityname:Language.OCR~~~sl-si~0.0.1.0 `
    /capabilityname:Language.OCR~~~sr-cyrl-rs~0.0.1.0 `
    /capabilityname:Language.OCR~~~sr-latn-rs~0.0.1.0 `
    /capabilityname:Language.OCR~~~sv-se~0.0.1.0 `
    /capabilityname:Language.OCR~~~tr-tr~0.0.1.0 `
    /capabilityname:Language.OCR~~~zh-cn~0.0.1.0 `
    /capabilityname:Language.OCR~~~zh-hk~0.0.1.0 `
    /capabilityname:Language.OCR~~~zh-tw~0.0.1.0 `
    /capabilityname:Language.Speech~~~da-dk~0.0.1.0 `
    /capabilityname:Language.Speech~~~de-de~0.0.1.0 `
    /capabilityname:Language.Speech~~~en-au~0.0.1.0 `
    /capabilityname:Language.Speech~~~en-ca~0.0.1.0 `
    /capabilityname:Language.Speech~~~en-gb~0.0.1.0 `
    /capabilityname:Language.Speech~~~en-in~0.0.1.0 `
    /capabilityname:Language.Speech~~~en-us~0.0.1.0 `
    /capabilityname:Language.Speech~~~es-es~0.0.1.0 `
    /capabilityname:Language.Speech~~~es-mx~0.0.1.0 `
    /capabilityname:Language.Speech~~~fr-ca~0.0.1.0 `
    /capabilityname:Language.Speech~~~fr-fr~0.0.1.0 `
    /capabilityname:Language.Speech~~~it-it~0.0.1.0 `
    /capabilityname:Language.Speech~~~ja-jp~0.0.1.0 `
    /capabilityname:Language.Speech~~~pt-br~0.0.1.0 `
    /capabilityname:Language.Speech~~~zh-cn~0.0.1.0 `
    /capabilityname:Language.Speech~~~zh-hk~0.0.1.0 `
    /capabilityname:Language.Speech~~~zh-tw~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~ar-eg~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~ar-sa~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~bg-bg~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~ca-es~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~cs-cz~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~da-dk~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~de-at~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~de-ch~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~de-de~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~el-gr~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~en-au~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~en-ca~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~en-gb~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~en-ie~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~en-in~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~en-us~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~es-es~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~es-mx~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~fi-fi~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~fr-ca~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~fr-ch~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~fr-fr~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~he-il~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~hi-in~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~hr-hr~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~hu-hu~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~id-id~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~it-it~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~ja-jp~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~ko-kr~0.0.1.0 `
```

```
    /capabilityname:Language.TextToSpeech~~~ms-my~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~nb-no~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~nl-be~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~nl-nl~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~pl-pl~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~pt-br~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~pt-pt~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~ro-ro~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~ru-ru~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~sk-sk~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~sl-si~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~sv-se~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~ta-in~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~th-th~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~tr-tr~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~vi-vn~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~zh-cn~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~zh-hk~0.0.1.0 `
    /capabilityname:Language.TextToSpeech~~~zh-tw~0.0.1.0 `
    /capabilityname:MathRecognizer~~~~0.0.1.0 `
    /capabilityname:Microsoft.Onecore.StorageManagement~~~~0.0.1.0 `
    /capabilityname:Microsoft.WebDriver~~~~0.0.1.0 `
    /capabilityname:Microsoft.Windows.MSPaint~~~~0.0.1.0 `
    /capabilityname:Microsoft.Windows.Notepad~~~~0.0.1.0 `
    /capabilityname:Microsoft.Windows.PowerShell.ISE~~~~0.0.1.0 `
    /capabilityname:Microsoft.Windows.StorageManagement~~~~0.0.1.0 `
    /capabilityname:Microsoft.Windows.WordPad~~~~0.0.1.0 `
    /capabilityname:Msix.PackagingTool.Driver~~~~0.0.1.0 `
    /capabilityname:NetFX3~~ `
    /capabilityname:Network.Irda~~~~0.0.1.0 `
    /capabilityname:OneCoreUAP.OneSync~~~~0.0.1.0 `
    /capabilityname:OpenSSH.Client~~~~0.0.1.0 `
    /capabilityname:OpenSSH.Server~~~~0.0.1.0 `
    /capabilityname:Print.EnterpriseCloudPrint~~~~0.0.1.0 `
    /capabilityname:Print.Fax.Scan~~~~0.0.1.0 `
    /capabilityname:Print.Management.Console~~~~0.0.1.0 `
    /capabilityname:Print.MopriaCloudService~~~~0.0.1.0 `
    /capabilityname:RasCMAK.Client~~~~0.0.1.0 `
    /capabilityname:RIP.Listener~~~~0.0.1.0 `
    /capabilityname:Rsat.ActiveDirectory.DS-LDS.Tools~~~~0.0.1.0 `
    /capabilityname:Rsat.BitLocker.Recovery.Tools~~~~0.0.1.0 `
    /capabilityname:Rsat.CertificateServices.Tools~~~~0.0.1.0 `
    /capabilityname:Rsat.DHCP.Tools~~~~0.0.1.0 `
    /capabilityname:Rsat.Dns.Tools~~~~0.0.1.0 `
    /capabilityname:Rsat.FailoverCluster.Management.Tools~~~~0.0.1.0 `
    /capabilityname:Rsat.FileServices.Tools~~~~0.0.1.0 `
    /capabilityname:Rsat.GroupPolicy.Management.Tools~~~~0.0.1.0 `
    /capabilityname:Rsat.IPAM.Client.Tools~~~~0.0.1.0 `
    /capabilityname:Rsat.LLDP.Tools~~~~0.0.1.0 `
    /capabilityname:Rsat.NetworkController.Tools~~~~0.0.1.0 `
    /capabilityname:Rsat.NetworkLoadBalancing.Tools~~~~0.0.1.0 `
    /capabilityname:Rsat.RemoteAccess.Management.Tools~~~~0.0.1.0 `
    /capabilityname:Rsat.RemoteDesktop.Services.Tools~~~~0.0.1.0 `
    /capabilityname:Rsat.ServerManager.Tools~~~~0.0.1.0 `
    /capabilityname:Rsat.Shielded.VM.Tools~~~~0.0.1.0 `
```

```
    /capabilityname:Rsat.StorageMigrationService.Management.Tools~~~~0.0.1.0
`
    /capabilityname:Rsat.StorageReplica.Tools~~~~0.0.1.0 `
    /capabilityname:Rsat.SystemInsights.Management.Tools~~~~0.0.1.0 `
    /capabilityname:Rsat.VolumeActivation.Tools~~~~0.0.1.0 `
    /capabilityname:Rsat.WSUS.Tools~~~~0.0.1.0 `
    /capabilityname:ServerCore.AppCompatibility~~~~0.0.1.0 `
    /capabilityname:SNMP.Client~~~~0.0.1.0 `
    /capabilityname:Tools.DeveloperMode.Core~~~~0.0.1.0 `
    /capabilityname:Tools.Graphics.DirectX~~~~0.0.1.0 `
    /capabilityname:Windows.Client.ShellComponents~~~~0.0.1.0 `
    /capabilityname:Windows.Desktop.EMS-SAC.Tools~~~~0.0.1.0 `
    /capabilityname:WMI-SNMP-Provider.Client~~~~0.0.1.0 `
    /capabilityname:XPS.Viewer~~~~0.0.1.0

    # This one is large, lets skip for now
    #/capabilityname:Analog.Holographic.Desktop~~~~0.0.1.0 `


# Copy language caps to the repo
Copy-Item -Path $OS_LP_PATH -Destination $REPO_PATH -Force -ErrorAction stop
| Out-Null

# Dismount OS image
Dismount-WindowsImage -Path $MAIN_OS_MOUNT -Discard -ErrorAction ignore |
Out-Null

# Dismount ISO images
Write-Host "Dismounting ISO images"
Dismount-DiskImage -ImagePath $LP_ISO_PATH -ErrorAction ignore | Out-Null
Dismount-DiskImage -ImagePath $FOD_ISO_PATH -ErrorAction ignore | Out-Null
```

Saving optional content in the source operating system

To save optional content state in the source operating system, we create a custom action script to run before the operating system installs. In this script, we save optional features and language resources to a file. We also make a local copy of the repo with only those files needed based on the languages installed on the source operating system. This action limits the files to copy.

PowerShell

```
$OUTPUT_PATH = "C:\TEMP\"
$LOG_PATH = $OUTPUT_PATH + "log.txt"
$OUTPUT_PATH = "C:\TEMP\"
$LOG_PATH = $OUTPUT_PATH + "log.txt"
$LANG_PATH = $OUTPUT_PATH + "sourceLang.txt"
$CAP_PATH = $OUTPUT_PATH + "sourceCapability.txt"
$OSVERSION_PATH = $OUTPUT_PATH + "sourceVersion.txt"
$REPO_PATH = "Z:\Repo\"
```

```
$LOCAL_REPO_PATH = $OUTPUT_PATH + "Local_Repo\"

Function Get-TS { return "{0:HH:mm:ss}" -f (Get-Date) }

Function Log
{
    param (
        [Parameter(Mandatory=$True)]
        [string]$MESSAGE
    )

    $M = "$(Get-TS): PreInstall: $MESSAGE"
    Write-Host $M
    Add-Content -Path $LOG_PATH -Value $M

 }

Function IsLangFile
{
    param (
        [Parameter(Mandatory=$True)]
        [string]$PATH
    )

    if (($PATH -match '[-_~]ar[-_~]') -or ($PATH -match '[-_~]bg[-_~]') -or
($PATH -match '[-_~]cs[-_~]') -or `
        ($PATH -match '[-_~]da[-_~]') -or ($PATH -match '[-_~]de[-_~]') -or
($PATH -match '[-_~]el[-_~]') -or `
        ($PATH -match '[-_~]en[-_~]') -or ($PATH -match '[-_~]es[-_~]') -or
($PATH -match '[-_~]et[-_~]') -or `
        ($PATH -match '[-_~]fi[-_~]') -or ($PATH -match '[-_~]fr[-_~]') -or
($PATH -match '[-_~]he[-_~]') -or `
        ($PATH -match '[-_~]hr[-_~]') -or ($PATH -match '[-_~]hu[-_~]') -or
($PATH -match '[-_~]it[-_~]') -or `
        ($PATH -match '[-_~]ja[-_~]') -or ($PATH -match '[-_~]ko[-_~]') -or
($PATH -match '[-_~]lt[-_~]') -or `
        ($PATH -match '[-_~]lv[-_~]') -or ($PATH -match '[-_~]nb[-_~]') -or
($PATH -match '[-_~]nl[-_~]') -or `
        ($PATH -match '[-_~]pl[-_~]') -or ($PATH -match '[-_~]pt[-_~]') -or
($PATH -match '[-_~]ro[-_~]') -or `
        ($PATH -match '[-_~]ru[-_~]') -or ($PATH -match '[-_~]sk[-_~]') -or
($PATH -match '[-_~]sl[-_~]') -or `
        ($PATH -match '[-_~]sv[-_~]') -or ($PATH -match '[-_~]th[-_~]') -or
($PATH -match '[-_~]tr[-_~]') -or `
        ($PATH -match '[-_~]uk[-_~]') -or ($PATH -match '[-_~]zh[-_~]') -or
($PATH -match '[-_~]sr[-_~]')) {
        return $True
    }
    else {
        return $False
    }
 }

# Remove the log
Remove-Item -Path $LOG_PATH -Force -ErrorAction ignore | Out-Null
```

```
Log "Starting"

# Remove state files, keep repo if it exists
Remove-Item -Path $LANG_PATH -Force -ErrorAction ignore | Out-Null
Remove-Item -Path $CAP_PATH -Force -ErrorAction ignore | Out-Null
Remove-Item -Path $OSVERSION_PATH -Force -ErrorAction ignore | Out-Null

# Get OS version, to use later for detecting compat scans versus OS
installation
$OSINFO = Get-CimInstance Win32_OperatingSystem
Log "OS Version: $($OSINFO.Version)"
Add-Content -Path $OSVERSION_PATH -Value $OSINFO.Version

# Get installed languages from international settings
$INTL = DISM.exe /Online /Get-Intl /English

# Save only output lines with installed languages
$LANGUAGES = $INTL | Select-String -SimpleMatch 'Installed language(s)'

# Replace with null so we have a simple list of language codes
$LANGUAGES = $LANGUAGES | ForEach-Object {$_.Line.Replace("Installed
language(s): ","")}

# Save System Language, save only output line with default system language
$SYSLANG = $INTL | Select-String -SimpleMatch 'Default system UI language'

# Replace with null so we have the language code
$SYSLANG = $SYSLANG | ForEach-Object {$_.Line.Replace("Default system UI
language : ","")}

# Save these languages
Log "Default system UI language on source OS: $($SYSLANG)"
ForEach ($ITEM in $LANGUAGES) {
    Log "Installed language on source OS: $($ITEM)"
    Add-Content -Path $LANG_PATH -Value $ITEM
}

# Get and save installed packages, we'll use this for debugging
$PACKAGES = Get-WindowsPackage -Online
ForEach ($ITEM in $PACKAGES) {
    if($ITEM.PackageState -eq "Installed") {
        Log "Package $($ITEM.PackageName) is installed"
    }
}

# Get and save capabilities
$CAPABILITIES = Get-WindowsCapability -Online
ForEach ($ITEM in $CAPABILITIES) {
    if($ITEM.State -eq "Installed") {
        Log "Capability $($ITEM.Name) is installed"
        Add-Content -Path $CAP_PATH -Value $ITEM.Name
    }
}

# Copy a subset of the Repo files locally, all neutral files and the
```

```
languages needed
$REPO_FILES = Get-ChildItem $REPO_PATH -file -Recurse
ForEach ($FILE in $REPO_FILES) {
    $PATH = ($FILE.DirectoryName + "\") -Replace
[Regex]::Escape($REPO_PATH), $LOCAL_REPO_PATH
    If (!(Test-Path $Path)) {
        New-Item -ItemType Directory -Path $PATH -Force | Out-Null
    }
    If ((IsLangFile $FILE.Name)) {

        # Only copy those files where we need the primary languages from the
source OS
        ForEach ($ITEM in $LANGUAGES) {
            if ($FILE.Name -match $Item) {

                If (!(Test-Path (Join-Path $Path $File.Name))) {
                    Copy-Item $FILE.FullName -Destination $PATH -Force
                    Log "Copied file $($FILE.FullName) to local repository"
                }
                else {
                    Log "File $($FILE.Name) already exists in local
repository"
                }
            }
        }
    } Else {

        # Copy all 'neutral files' and those language specific that are not
in the core 38
        If (!(Test-Path (Join-Path $Path $File.Name))) {
            Copy-Item $FILE.FullName -Destination $PATH -Force
            Log "Copied file $($FILE.FullName) to local repository"
        }
        else {
            Log "File $($FILE.Name) already exists in local repository"
        }
    }
}

Log ("Exiting")
```

Adding optional content in the target operating system

After setup has completed successfully, we use success.cmd to retrieve the optional content state from the source operating system and install in the new operating system only if that's missing. Then, apply the latest monthly update as a final step.

PowerShell

```
$OUTPUT_PATH = "C:\TEMP\"
$LOG_PATH = $OUTPUT_PATH + "log.txt"
$LANG_PATH = $OUTPUT_PATH + "sourceLang.txt"
$CAP_PATH = $OUTPUT_PATH + "sourceCapability.txt"
$OSVERSION_PATH = $OUTPUT_PATH + "sourceVersion.txt"
$LOCAL_REPO_PATH = $OUTPUT_PATH + "Local_Repo\"
$LCU_PATH = $OUTPUT_PATH + "Windows10.0-KB4565503-x64_PSFX.cab"
$PENDING = $false

Function Get-TS { return "{0:HH:mm:ss}" -f (Get-Date) }

Function Log
{
    param (
        [Parameter(Mandatory=$True)]
        [string]$MESSAGE
    )

    $M = "$(Get-TS): PostInstall: $MESSAGE"
    Write-Host $M
    Add-Content -Path $LOG_PATH -Value $M

 }

Log "Starting"

# Get OS version
$OSINFO = Get-CimInstance Win32_OperatingSystem
Log "OS Version: $($OSINFO.Version)"

# Check for source OS state, just to be sure
if (!(Test-Path $LANG_PATH) -or !(Test-Path $CAP_PATH) -or !(Test-Path
$OSVERSION_PATH) ) {
    Log "Source OS state is missing."
}

# If this script is executing and the OS version hasn't changed, let's exit
out.
else {

    # Retrive OS version from source OS
    $SOURCE_OSVERSION  = Get-Content -Path $OSVERSION_PATH
    if ($OSINFO.Version -eq $SOURCE_OSVERSION) {
        Log "OS Version hasn't changed."
    }

    else {

        # Retrive language list from source OS
        $SOURCE_LANGUAGES  = Get-Content -Path $LANG_PATH

        # Get installed languages from International Settings
        $INTL = DISM.exe /Online /Get-Intl /English
```

```
        # Save System Language, save only output line with default system
language
        $SYS_LANG = $INTL | Select-String -SimpleMatch 'Default system UI
language'

        # Replace with null so we have the language code
        $SYS_LANG = $SYS_LANG | ForEach-Object {$_.Line.Replace("Default
system UI language : ","")}

        # Get and save installed packages, we'll use this for debugging
        $PACKAGES = Get-WindowsPackage -Online
        ForEach ($ITEM in $PACKAGES) {
            if($ITEM.PackageState -eq "Installed") {
                Log "Package $($ITEM.PackageName) is installed"
            }
        }

        # Loop through source OS languages, and install if missing on target
OS
        ForEach ($SOURCE_ITEM in $SOURCE_LANGUAGES) {
            if ($SOURCE_ITEM -ne $SYS_LANG) {

                # add missing languages except the system language
                Log "Adding language Microsoft-Windows-Client-Language-
Pack_x64_$($SOURCE_ITEM).cab"
                try {
                    Add-WindowsPackage -Online -PackagePath
"$($LOCAL_REPO_PATH)\Microsoft-Windows-Client-Language-
Pack_x64_$($SOURCE_ITEM).cab" -ErrorAction stop | Out-Null
                }
                catch {
                    Log $_.Exception.Message
                }
            }
        }

        # Retrieve capabilities from source OS and target OS
        $SOURCE_CAPABILITIES  = Get-Content -Path $CAP_PATH
        $CAPABILITIES = Get-WindowsCapability -Online

        # Loop through source OS capabilities, and install if missing on
target OS
        ForEach ($SOURCE_ITEM in $SOURCE_CAPABILITIES) {
            $INSTALLED = $false
            ForEach ($ITEM in $CAPABILITIES) {
                if ($ITEM.Name -eq $($SOURCE_ITEM)) {
                    if ($ITEM.State -eq "Installed") {
                        $INSTALLED = $true
                        break
                    }
                }
            }

            # Add if not already installed
            if (!($INSTALLED)) {
```

```
                Log "Adding capability $SOURCE_ITEM"
                try {
                    Add-WindowsCapability -Online -Name $SOURCE_ITEM -Source
$LOCAL_REPO_PATH -ErrorAction stop | Out-Null
                }
                catch {
                    Log $_.Exception.Message
                }
            }
            else {
                Log "Capability $SOURCE_ITEM is already installed"
            }
        }

        # Add LCU, this is required after adding FODs and languages
        Log ("Adding LCU")
        Add-WindowsPackage -Online -PackagePath $LCU_PATH -NoRestart

        # Get packages, we'll use this for debugging and to see if we need
to restart to install
        $PACKAGES = Get-WindowsPackage -Online
        ForEach ($ITEM in $PACKAGES) {
            Log "Package $($ITEM.PackageName) is $($ITEM.PackageState)"
            if ($ITEM.PackageState -eq "InstallPending") {
                $PENDING = $true
            }
        }
    }
}

# Remove local repository and state files
Remove-Item -Path $LANG_PATH -Force -ErrorAction ignore | Out-Null
Remove-Item -Path $CAP_PATH -Force -ErrorAction ignore | Out-Null
Remove-Item -Path $OSVERSION_PATH -Force -ErrorAction ignore | Out-Null
Remove-Item -Path $LOCAL_REPO_PATH -Force -Recurse -ErrorAction ignore |
Out-Null

# Restarting the computer to let setup process to exit cleanly
if ($PENDING) {
    Log ("Install pending packages exists, restarting in 10 seconds")
    Start-Process -FilePath cmd -ArgumentList "/C shutdown /r /t 10 /f"
}

Log ("Exiting")
```

Feedback

Was this page helpful? Yes No

Provide product feedback

Deploy feature updates for user-initiated installations

Article • 01/19/2023 • Applies to: ✅ Windows 10, ✅ Microsoft Configuration Manager

Use the following steps to deploy a feature update for a user-initiated installation.

Get ready to deploy feature updates

Step 1: Enable peer cache

Use **peer cache** to help manage deployment of content to clients in remote locations. Peer cache is a built-in Configuration Manager solution that enables clients to share content with other clients directly from their local cache.

Enable Configuration Manager client in full OS to share content if you have clients in remote locations that would benefit from downloading feature update content from a peer instead of downloading it from a distribution point (or Microsoft Update).

Step 2: Override the default Windows setup priority (Windows 10, version 1709 and later)

If you're deploying **Feature update to Windows 10, version 1709** or later, by default, portions of setup are configured to run at a lower priority. This can result in a longer total install time for the feature update. When deploying within a maintenance window, we recommend that you override this default behavior to benefit from faster total install times. To override the default priority, create a file called SetupConfig.ini on each machine to be upgraded in the below location containing the single section noted.

%systemdrive%\Users\Default\AppData\Local\Microsoft\Windows\WSUS\SetupConfig.ini

```ini
[SetupConfig]
Priority=Normal
```

You can use the new Run Scripts feature to run a PowerShell script like the sample below to create the SetupConfig.ini on target devices.

PowerShell

```
#Parameters
Param(
  [string] $PriorityValue = "Normal"
  )

#Variable for ini file path
$iniFilePath =
"$env:SystemDrive\Users\Default\AppData\Local\Microsoft\Windows\WSUS\SetupCo
nfig.ini"

#Variables for SetupConfig
$iniSetupConfigSlogan = "[SetupConfig]"
$iniSetupConfigKeyValuePair =@{"Priority"=$PriorityValue;}

#Init SetupConfig content
$iniSetupConfigContent = @"
$iniSetupConfigSlogan
"@

#Build SetupConfig content with settings
foreach ($k in $iniSetupConfigKeyValuePair.Keys)
{
    $val = $iniSetupConfigKeyValuePair[$k]

    $iniSetupConfigContent =
$iniSetupConfigContent.Insert($iniSetupConfigContent.Length, "`r`n$k=$val")
}

#Write content to file
New-Item $iniFilePath -ItemType File -Value $iniSetupConfigContent -Force

<#
Disclaimer
Sample scripts are not supported under any Microsoft standard support
program or service. The sample scripts is
provided AS IS without warranty of any kind. Microsoft further disclaims all
implied warranties including, without
limitation, any implied warranties of merchantability or of fitness for a
particular purpose. The entire risk
arising out of the use or performance of the sample script and documentation
remains with you. In no event shall
Microsoft, its authors, or anyone else involved in the creation, production,
or delivery of the scripts be liable
for any damages whatsoever (including, without limitation, damages for loss
of business profits, business interruption,
loss of business information, or other pecuniary loss) arising out of the
use of or inability to use the sample script
or documentation, even if Microsoft has been advised of the possibility of
such damages.
#>
```

> **Note**
>
> If you elect not to override the default setup priority, you will need to increase the maximum run time value for Feature Update to Windows 10, version 1709 or higher from the default of 60 minutes. A value of 240 minutes may be required. Remember to ensure that your maintenance window duration is larger than your defined maximum run time value.

Manually deploy feature updates in a user-initiated installation

The following sections provide the steps to manually deploy a feature update.

Step 1: Specify search criteria for feature updates

There are potentially a thousand or more feature updates displayed in the Configuration Manager console. The first step in the workflow for manually deploying a feature update is to identify the feature updates that you want to deploy.

1. In the Configuration Manager console, select **Software Library**.
2. In the Software Library workspace, expand **Windows 10 Servicing**, and select **All Windows 10 Updates**. The synchronized feature updates are displayed.
3. In the search pane, filter to identify the feature updates that you need by using one or both of the following steps:
   - In the **search** text box, type a search string that filters for the feature updates. For example, type the version number for a specific feature update, or enter a string that would appear in the title of the feature update.
   - Select **Add Criteria**, select the criteria that you want to use to filter software updates, select **Add**, and then provide the values for the criteria. For example, Title contains 1803, **Required** is greater than or equal to 1, and **Language** equals English.
4. Save the search for future use.

Step 2: Download the content for the feature update

Before you deploy the feature updates, you can download the content as a separate step. Do this download so you can verify that the content is available on the distribution points before you deploy the feature updates. Downloading first helps you avoid any unexpected issues with the content delivery. Use the following procedure to download the content for feature updates before creating the deployment.

1. In the Configuration Manager console, navigate to **Software Library** > **Windows 10 Servicing**.

2. Choose the feature update(s) to download by using your saved search criteria. Select one or more of the feature updates returned, right-click, and select **Download**.

   The **Download Software Updates Wizard** opens.

3. On the **Deployment Package** page, configure the following settings: **Create a new deployment package**: Select this setting to create a new deployment package for the software updates that are in the deployment. Configure the following settings:

   - **Name**: Specifies the name of the deployment package. The package must have a unique name that briefly describes the package content. It's limited to 50 characters.

   - **Description**: Specifies the description of the deployment package. The package description provides information about the package contents and is limited to 127 characters.

   - **Package source**: Specifies the location of the feature update source files. Type a network path for the source location, for example, \\server\sharename\path, or select **Browse** to find the network location. You must create the shared folder for the deployment package source files before you proceed to the next page.

     > ⓘ **Important**
     > - The deployment package source location that you specify cannot be used by another software deployment package.
     > - The SMS Provider computer account and the user that is running the wizard to download the feature updates must both have Write NTFS permissions on the download location. You should carefully restrict access to the download location to reduce the risk of attackers tampering with the feature update source files.

- You can change the package source location in the deployment package properties after Configuration Manager creates the deployment package. But if you do so, you must first copy the content from the original package source to the new package source location.

Select **Next**.

4. On the **Distribution Points** page, specify the distribution points or distribution point groups that will host the feature update files, and then select **Next**. For more information about distribution points, see Distribution point configurations.

> **Note**
>
> The Distribution Points page is available only when you create a new software update deployment package.

5. On the **Distribution Settings** page, specify the following settings:

   - **Distribution priority**: Use this setting to specify the distribution priority for the deployment package. The distribution priority applies when the deployment package is sent to distribution points at child sites. Deployment packages are sent in priority order: **High**, **Medium**, or **Low**. Packages with identical priorities are sent in the order in which they were created. If there's no backlog, the package processes immediately regardless of its priority. By default, packages are sent using Medium priority.

   - **Enable for on-demand distribution**: Use this setting to enable on-demand content distribution to preferred distribution points. When this setting is enabled, the management point creates a trigger for the distribution manager to distribute the content to all preferred distribution points when a client requests the content for the package and the content isn't available on any preferred distribution points. For more information about preferred distribution points and on-demand content, see Content source location scenarios.

   - **Prestaged distribution point settings**: Use this setting to specify how you want to distribute content to prestaged distribution points. Choose one of the following options:
     - **Automatically download content when packages are assigned to distribution points**: Use this setting to ignore the prestage settings and

distribute content to the distribution point.
    - **Download only content changes to the distribution point**: Use this setting to prestage the initial content to the distribution point, and then distribute content changes to the distribution point.
    - **Manually copy the content in this package to the distribution point**: Use this setting to always prestage content on the distribution point. This setting is the default.

    For more information about prestaging content to distribution points, see Use Prestaged content. Select **Next**.

6. On the **Download Location** page, specify location that Configuration Manager will use to download the software update source files. As needed, use the following options:

    - **Download software updates from the Internet**: Select this setting to download the software updates from the location on the Internet. This is the default setting.

    - **Download software updates from a location on the local network**: Select this setting to download software updates from a local folder or shared network folder. Use this setting when the computer running the wizard doesn't have Internet access.

    > ⓘ **Note**
    >
    > When you use this setting, download the software updates from any computer with Internet access, and then copy the software updates to a location on the local network that is accessible from the computer running the wizard.

    Select **Next**.

7. On the **Language Selection** page, specify the languages for which the selected feature updates are to be downloaded, and then select **Next**. Ensure that your language selection matches the language(s) of the feature updates selected for download. For example, if you selected English and German based feature updates for download, select those same languages on the language selection page.

8. On the **Summary** page, verify the settings that you selected in the wizard, and then select **Next** to download the software updates.

9. On the **Completion** page, verify that the software updates were successfully downloaded, and then select **Close**.

To monitor content status

1. To monitor the content status for the feature updates, select **Monitoring** in the Configuration Manager console.
2. In the Monitoring workspace, expand **Distribution Status**, and then select **Content Status**.
3. Select the feature update package that you previously identified to download the feature updates.
4. On the **Home** tab, in the Content group, select **View Status**.

Step 3: Deploy the feature update

After you determine which feature updates you intend to deploy, you can manually deploy the feature update(s). Use the following procedure to manually deploy the feature update(s).

1. In the Configuration Manager console, select **Software Library**.

2. In the Software Library workspace, expand **Windows 10 Servicing**, and select **All Windows 10 Updates**.

3. Choose the feature update(s) to deploy by using your saved search criteria. Select one or more of the feature updates returned, right select, and select **Deploy**.

   The **Deploy Software Updates Wizard** opens.

4. On the General page, configure the following settings:

   - **Name**: Specify the name for the deployment. The deployment must have a unique name that describes the purpose of the deployment and differentiates it from other deployments in the Configuration Manager site. By default, Configuration Manager automatically provides a name for the deployment in the following format: **Microsoft Software Updates - <date> <time>**
   - **Description**: Specify a description for the deployment. The description provides an overview of the deployment and any other relevant information that helps to identify and differentiate the deployment among others in Configuration Manager site. The description field is optional, has a limit of 256 characters, and has a blank value by default.

- **Software Update/Software Update Group**: Verify that the displayed software update group, or software update, is correct.
- **Select Deployment Template**: Specify whether to apply a previously saved deployment template. You can configure a deployment template to contain multiple common software update deployment properties and then apply the template when you deploy subsequent software updates to ensure consistency across similar deployments and to save time.
- **Collection**: Specify the collection for the deployment, as applicable. Members of the collection receive the feature updates that are defined in the deployment.

5. On the Deployment Settings page, configure the following settings:

- **Type of deployment**: Specify the deployment type for the software update deployment. Select **Required** to create a mandatory software update deployment in which the feature updates are automatically installed on clients before a configured installation deadline.

> **Important**
>
> After you create the software update deployment, you cannot later change the type of deployment.

> **Note**
>
> A software update group deployed as **Required** will be downloaded in background and honor BITS settings, if configured.

- **Use Wake-on-LAN to wake up clients for required deployments**: Specify whether to enable Wake On LAN at the deadline to send wake-up packets to computers that require one or more software updates in the deployment. Any computers that are in sleep mode at the installation deadline time will be awakened so the software update installation can initiate. Clients that are in sleep mode that don't require any software updates in the deployment aren't started. By default, this setting isn't enabled and is available only when **Type of deployment** is set to **Required**.

> **Warning**

> Before you can use this option, computers and networks must be configured for Wake On LAN.

- **Detail level**: Specify the level of detail for the state messages that are reported by client computers.

6. On the Scheduling page, configure the following settings:

   - **Schedule evaluation**: Specify whether the available time and installation deadline times are evaluated according to UTC or the local time of the computer running the Configuration Manager console.

   - **Software available time**: Select **Specific time** to specify when the software updates will be available to clients:
     - **Specific time**: Select this setting to make the feature update in the deployment available to clients at a specific date and time. Specify a date and time that corresponds with the start of your fixed servicing window. When the deployment is created, the client policy is updated and clients are made aware of the deployment at their next client policy polling cycle. However, the feature update in the deployment isn't available for installation until after the specified date and time are reached and the required content has been downloaded.

   - **Installation deadline**: Select **Specific time** to specify the installation deadline for the software updates in the deployment.

     > ⓘ **Note**
     >
     > You can configure the installation deadline setting only when **Type of deployment** is set to **Required** on the Deployment Settings page.

     - **Specific time**: Select this setting to automatically install the software updates in the deployment at a specific date and time. However, for the purposes of the fixed servicing window, set the installation deadline date and time to a future value, well beyond the fixed servicing window.

     Required deployments for software updates can benefit from functionality called advanced download. When the software available time is reached, clients start downloading the content based on a randomized time. The feature update won't be displayed in Software Center for installation until the content is fully downloaded. This ensures that the feature update installation starts immediately when initiated.

7. On the User Experience page, configure the following settings:

- **User notifications**: Specify **Display in Software Center and show all notifications**.

- **Deadline behavior**: Available only when **Type of deployment** is set to **Required** on the Deployment Settings page. Specify the behavior that is to occur when the deadline is reached for the software update deployment. Specify whether to install the software updates in the deployment. Also specify whether to perform a system restart after software update installation regardless of a configured maintenance window.

  > **Note**
  >
  > Remember that the installation deadline date and time will be well into the future to allow plenty of time for the user-initiated install during a fixed servicing window.

- **Device restart behavior**: Available only when **Type of deployment** is set to **Required** on the Deployment Settings page. Specify whether to suppress a system restart on servers and workstations after software updates are installed and a system restart is required to complete the installation.

  > **Important**
  >
  > Suppressing system restarts can be useful in server environments or for cases in which you do not want the computers that are installing the software updates to restart by default. However, doing so can leave computers in an insecure state, whereas allowing a forced restart helps to ensure immediate completion of the software update installation.

- **Write filter handling for Windows Embedded devices**: When you deploy software updates to Windows Embedded devices that are write filter enabled, you can specify to install the software update on the temporary overlay and either commit changes later or commit the changes at the installation deadline or during a maintenance window. When you commit changes at the installation deadline or during a maintenance window, a restart is required and the changes persist on the device.

  > **Note**

> When you deploy a software update to a Windows Embedded device, make sure that the device is a member of a collection that has a configured maintenance window.

- **Software updates deployment re-evaluation behavior upon restart**: Starting in Configuration Manager version 1606, select this setting to configure software updates deployments to have clients run a software updates compliance scan immediately after a client installs software updates and restarts. This enables the client to check for additional software updates that become applicable after the client restarts, and to then install them (and become compliant) during the same maintenance window.

8. On the Alerts page, configure how Configuration Manager and System Center Operations Manager generate alerts for this deployment. You can configure alerts only when **Type of deployment** is set to **Required** on the Deployment Settings page.

> ⓘ **Note**
>
> You can review recent software updates alerts from the **Software Updates** node in the **Software Library** workspace.

9. On the Download Settings page, configure the following settings:

   - Specify whether the client will download and install the software updates when a client is connected to a slow network or is using a fallback content location.

   - Specify whether to have the client download and install the software updates from a fallback distribution point when the content for the software updates isn't available on a preferred distribution point.

   - **Allow clients to share content with other clients on the same subnet**: Specify whether to enable the use of BranchCache for content downloads. For more information about BranchCache, see Fundamental concepts for content management.

   - **If software updates are not available on distribution point in current, neighbor or site groups, download content from Microsoft Updates**: Select this setting to have clients that are connected to the intranet download software updates from Microsoft Update if software updates aren't available

on distribution points. Internet-based clients can always go to Microsoft Update for software updates content.

- Specify whether to allow clients to download after an installation deadline when they use metered Internet connections. Internet providers sometimes charge by the amount of data that you send and receive when you are on a metered Internet connection.

> **Note**
>
> Clients request the content location from a management point for the software updates in a deployment. The download behavior depends upon how you have configured the distribution point, the deployment package, and the settings on this page. For more information, see Content source location scenarios.

10. On the Summary page, review the settings. To save the settings to a deployment template, select **Save As Template**, enter a name and select the settings that you want to include in the template, and then select **Save**. To change a configured setting, select the associated wizard page and change the setting.

11. Select **Next** to deploy the feature update(s).

Step 4: Monitor the deployment status

After you deploy the feature update(s), you can monitor the deployment status. Use the following procedure to monitor the deployment status:

1. In the Configuration Manager console, navigate to **Monitoring** > **Overview** > **Deployments**.
2. Select the software update group or software update for which you want to monitor the deployment status.
3. On the **Home** tab, in the **Deployment** group, select **View Status**.

Feedback

Was this page helpful? Yes No

Provide product feedback

Safeguard holds

Applies to: ✅ Windows 11, ✅ Windows 10

Microsoft uses quality and compatibility data to identify issues that might cause a Windows client feature update to fail or roll back. When we find such an issue, we might apply safeguard holds to the updating service to prevent affected devices from installing the update in order to safeguard them from these experiences. We also use safeguard holds when a customer, a partner, or Microsoft internal validation finds an issue that would cause severe effect (for example, rollback of the update, data loss, loss of connectivity, or loss of key functionality) and when a workaround isn't immediately available.

Safeguard holds prevent a device with a known issue from being offered a new operating system version. We renew the offering once a fix is found and verified. We use holds to ensure customers have a successful experience as their device moves to a new version of Windows client.

The safeguard holds lifespan varies depending on the time required to investigate and fix an issue. During this time, Microsoft works diligently to procure, develop, and validate a fix and then offer it to affected devices. We monitor quality and compatibility data to confirm that a fix is complete before releasing the safeguard hold. Once we release the safeguard hold, Windows Update resumes offering new operating system versions to devices.

Safeguard holds only affect devices that use the Windows Update service for updates. We encourage IT admins who manage updates to devices through other channels (such as media installations or updates coming from Windows Server Update Services) to remain aware of known issues that might also be present in their environments.

IT admins managing updates using Windows Autopatch also benefit from safeguard holds on devices that are likely to be affected by an issue. To learn more, see Safeguard holds against likely and known issues.

Am I affected by a safeguard hold?

IT admins can use Windows Update for Business reports to monitor various update health metrics for devices in their organization. The reports provide a list of active safeguard holds to provide you with insight into the safeguard holds that are preventing devices from updating or upgrading.

Windows Update for Business reports identifies safeguard holds by their 8-digit identifiers. For safeguard holds associated with publicly discussed known issues, you can find more details

about the issue on the Windows release health dashboard by searching for the safeguard hold ID on the **Known issues** page for the relevant release.

On devices that use Windows Update (but not Windows Update client policies), the **Windows Update** page in the Settings app displays a message stating that an update is on its way, but not ready for the device. Instead of the option to download and install the update, users see a message.

This message means that the device is protected by one or more safeguard holds. When the issue is resolved and the update is safe to install, we release the safeguard hold so the update can resume safely.

What can I do?

We recommend that you don't attempt to manually update until issues have been resolved and holds released.

> ⊗ **Caution**
>
> Opting out of a safeguard hold can put devices at risk from known performance issues. We strongly recommend that you complete robust testing to ensure the impact is acceptable before opting out.

With that in mind, IT admins who stay informed with Windows Update for Business reports and the Windows release health dashboard can choose to temporarily opt-out of the protection of all safeguard holds and allow an update to proceed. We recommend opting out only in an IT environment and for validation purposes. If you do opt out of a hold, this condition is temporary. Once an update is complete, the protection of safeguard holds is reinstated automatically.

Troubleshoot a safeguard hold manually

The gated status (GStatus) registry value indicates if a safeguard hold is in effect or not.

The GStatus value can be found in the following registry key:
`HKLM:\SOFTWARE\Microsoft\Windows NT\CurrentVersion\AppCompatFlags\Appraiser\GWX`

- **GStatus** = `2`: A safeguard hold isn't in effect
- **GStatus** = `0`: A safeguard hold is in effect

If a safeguard hold is in effect (**GStatus** = `0`), you can also determine the safeguard ID so you can get additional information about the hold. Use the following registry key, changing the corresponding OS version to match the version of Windows the device uses:

`HKLM:\SOFTWARE\Microsoft\Windows NT\CurrentVersion\AppCompatFlags\TargetVersionUpgradeExperienceIndicators\GE24H2`

Expand table

Value name	Value	Description
GStatus	0	A safeguard hold is in effect
GatedBlockId	52796844	The safeguard ID
GatedBlockReason	Other	The reason for the safeguard hold

Once you have the safeguard ID, you can find more information about it using the Windows release health dashboard. For example, safeguard ID `52796844` is in the Resolved issues for Windows 11, version 24H2 article and was applied for a fingerprint sensor issue. For more information about finding the safeguard ID, see the Access safeguard hold details with Update Compliance blog post.

In some instances, an outdated safeguard hold might still be in effect on the device. The example safeguard ID, `52796844`, would be outdated since the issue is resolved. An outdated hold can happen if the mechanism to update this information on the Windows device isn't working correctly. An outdated hold is most commonly caused by SSL inspection being enabled for one or more of the following endpoints:

Expand table

Area	Description	Protocol	Destination
Windows Update	The following endpoint is used for compatibility database updates for Windows.	HTTPS	adl.windows.com
Windows Update	The following endpoint is used for compatibility database updates for	HTTPS	adl.windows.com

Area	Description	Protocol	Destination
	Windows.		
Settings	The following endpoint is used as a way for apps to dynamically update their configuration. Apps such as System Initiated User Feedback and the Xbox app use it. If you turn off traffic for this endpoint, an app that uses this endpoint might stop working.	TLSv1.2/HTTPS/HTTP	settings-win.data.microsoft.com
Settings	The following endpoint is used as a way for apps to dynamically update their configuration. Apps such as System Initiated User Feedback and the Xbox app use this endpoint. If you turn off traffic for this endpoint, an app that uses this endpoint might stop working.	HTTPS	settings.data.microsoft.com

For more information, see Manage connection endpoints for Windows 11 Enterprise.

Last updated on 08/15/2025

Opt out of safeguard holds

Article • 08/23/2023 • Applies to: ✅ Windows 11, ✅ Windows 10

Safeguard holds prevent a device with a known compatibility issue from being offered a new Windows client feature update by using Windows Update. We use safeguard holds to protect the device and user from a failed or poor update experience. We renew the offering once a fix is issued and is verified on an affected device. For more information about safeguard holds, see Safeguard holds.

How can I opt out of safeguard holds?

IT admins can, if necessary, opt devices out of safeguard protections by using the disable safeguards policy. In a Mobile Device Management (MDM) tool, use the **Update/DisableWUfBSafeguards** CSP. In Group Policy, use the **Disable safeguards for Feature Updates** Group Policy. This policy is available to Windows Update client policies devices running the following operating systems:

- Windows 11
- Windows 10, version 1809, or later, with the October 2020 security update.

> ⊗ **Caution**
>
> Opting out of a safeguard hold can put devices at risk from known performance issues.

We recommend opting out only in an IT environment and for validation purposes. You can also validate an upcoming Windows client feature update version without the safeguards being applied by using the Release Preview channel of the Windows Insider Program for Business.

Disabling safeguards doesn't guarantee your device will be able to successfully update. The update might still fail and will likely result in a bad experience since you're bypassing the protection against known issues.

> ⓘ **Note**
>
> After a device installs a new Windows client version, the **Disable safeguards for Feature Updates** Group Policy will revert to **Not configured** even if it was

previously enabled. We do this to ensure the admin is consciously disabling Microsoft's default protection from known issues for each new feature update.

Feedback

Was this page helpful? Yes No

Provide product feedback

Manage device restarts after updates

Applies to: ✅ Windows 11, ✅ Windows 10

> **Looking for consumer information?** See Windows Update: FAQ

You can use group policy settings, mobile device management (MDM), or the Windows registry to configure when devices will restart after a Windows update is installed. You can schedule update installation and set policies for restart, configure active hours for when restarts shouldn't occur, or you can do both.

> **Note**
>
> Directly editing the Windows registry isn't recommended.

Schedule update installation

In group policy, within **Configure Automatic Updates**, you can configure a forced restart after a specified installation time.

To set the time, go to **Configure Automatic Updates**, select option **4 - Auto download and schedule the install**, and then use **Scheduled install time** to enter a time. Alternatively, you can specify that installation occurs during the automatic maintenance time. To configure this alternative method, use **Computer Configuration\Administrative Templates\Windows Components\Maintenance Scheduler**.

The setting to **Always automatically restart at the scheduled time** forces a restart after the specified installation time. It lets you configure a timer to warn a signed-in user that a restart is going to occur. This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

While not recommended, you can achieve the same result with the Windows registry. Under `HKLM\Software\Policies\Microsoft\Windows\WindowsUpdate\AU`, set `AuOptions` to `4` and set the install time with `ScheduledInstallTime`. Enable `AlwaysAutoRebootAtScheduledTime` and specify the delay in minutes through `AlwaysAutoRebootAtScheduledTimeMinutes`. Similar to group policy, `AlwaysAutoRebootAtScheduledTimeMinutes` sets the timer to warn a signed-in user that a restart is going to occur.

For a detailed description of these registry keys, see Registry keys used to manage restart.

Delay automatic restart

When you enable **Configure Automatic Updates** in group policy, you can also enable one of the following policies to delay an automatic restart after update installation:

- **Turn off auto-restart for updates during active hours** prevents automatic restart during active hours.
- **No auto-restart with logged on users for scheduled automatic updates installations** prevents automatic restart when a user is signed in. If a user schedules the restart in the update notification, the device restarts at the time the user specifies even if a user is signed in at the time. This policy only applies when **Configure Automatic Updates** is set to option **4 - Auto download and schedule the install**.

> **Note**
>
> - When using Remote Desktop Protocol (RDP) connections, only active RDP sessions are considered signed-in users. Devices that don't have locally signed-in users, or active RDP sessions, are restarted.
> - The **No auto-restart with logged on users for scheduled automatic updates installation** policy was never created as a CSP. In Group Policy this policy doesn't work exactly as per description. This policy can result in no quality update reboots period, given many users never log off. The recommendation to replace this would be to leverage compliance deadline and then to configure no-auto reboot to prevent non-user aware reboots prior to the deadline being reached. For server devices, leverage Configure Automatic Updates options 7 - notify to install and notify to reboot.

You can also use the Windows registry, to prevent automatic restarts when a user is signed in. Under `HKLM\Software\Policies\Microsoft\Windows\WindowsUpdate\AU`, set `AuOptions` to `4` and enable `NoAutoRebootWithLoggedOnUsers`. As with group policy, if a user schedules the restart in the update notification, it overrides this setting.

For a detailed description of these registry keys, see Registry keys used to manage restart.

Configure active hours

Active hours identify the period of time when you expect the device to be in use. Automatic restarts after an update occur outside of the active hours.

By default, active hours are from 8 AM to 5 PM on PCs. Users can manually change the active hours.

You can also specify the max active hours range. The specified range is counted from the active hours start time.

> **Note**
>
> The max active hours length for Windows 10, version 1607 and Windows Server 2016 is 12. Later versions support max active hours length of 18 hours.

Configure active hours with group policy

To configure active hours using group policy, go to **Computer Configuration\Administrative Templates\Windows Components\Windows Update** and open the **Turn off auto-restart for updates during active hours** policy setting. When the policy is enabled, you can set the start and end times for active hours.

Configure active hours with MDM

To configure active hours, MDM uses the following settings in the Update Policy CSP:

- ActiveHoursStart
- ActiveHoursEnd
- ActiveHoursMaxRange

Configure active hours through the Windows registry

This method isn't recommended, and should only be used when you can't use group policy or MDM. Any settings configured through the registry might conflict with any existing configuration that uses any of the other methods.

Configure active hours by setting a combination of the following registry values:

Under `HKLM\Software\Policies\Microsoft\Windows\WindowsUpdate` use `SetActiveHours` to enable or disable active hours and `ActiveHoursStart` and `ActiveHoursEnd` to specify the range of active hours.

For a detailed description of these registry keys, see Registry keys used to manage restart.

> **Tip**
>
> To manually configure active hours on a device, go to **Settings** > **Windows Update** > **Advanced options** and select **Active hours**.

Configure active hours maximum range

You can specify the maximum active hours range that users can set. This option gives you flexibility to leave some of the decision for active hours on the user's side, while making sure you allow enough time for updates to install. The maximum range is calculated from the active hours start time.

To configure the maximum range for active hours through group policy, go to **Computer Configuration\Administrative Templates\Windows Components\Windows Update** and open the setting to **Specify active hours range for auto-restarts**.

To configure the maximum range for active hours through MDM, use ActiveHoursMaxRange.

Limit restart delays

After Windows installs an update, it attempts to automatically restart outside of active hours. If the restart doesn't succeed after a default period of seven days, the user sees a notification

that a restart is required. To change the delay, use the setting to **Specify deadline before auto-restart for update installation**. The minimum value is two days and the maximum value is two weeks (14 days). This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Control restart notifications

Display options for update notifications

You can define which Windows Update notifications are displayed to the user. This policy doesn't control how and when updates are downloaded and installed.

To configure this behavior through group policy, go to **Computer Configuration\Administrative Templates\Windows Components\Windows Update** and select the policy for **Display options for update notifications**. Configure the following values:

- `0` (default): Use the default Windows Update notifications.
- `1`: Turn off most notifications but keep restart warnings.
- `2`: Turn off all notifications including restart warnings.

To configure this behavior through MDM, use UpdateNotificationLevel.

Starting in Windows 11, version 22H2, **Apply only during active hours** was added as another option for **Display options for update notifications**. When you select **Apply only during active hours**, the notifications are only disabled during active hours when you use options `1` or `2`. To ensure that the device stays updated, a notification is still shown during active hours if you select **Apply only during active hours**, and once a deadline is reached when you configure Specify deadlines for automatic updates and restarts.

To configure this behavior through MDM, use UpdateNotificationLevel.

Automatic restart notifications

You can override the default behavior for the automatic restart required notification. By default, this notification dismisses automatically.

- To configure this behavior through group policy, go to **Computer Configuration\Administrative Templates\Windows Components\Windows Update** and select the policy to **Configure auto-restart required notification for updates**. When configured to **2 - User Action**, a user that gets this notification must manually dismiss it. This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

- To configure this behavior through MDM, use AutoRestartRequiredNotificationDismissal. This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

You can also configure the period before an update that this notification shows up. The default value is 15 minutes.

- To change it through group policy, select **Configure auto-restart-reminder notifications for updates** under **Computer Configuration\Administrative Templates\Windows Components\Windows Update** and select the period in minutes. This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

- To change it through MDM, use AutoRestartNotificationSchedule. This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

In some cases, you don't need a notification to show up.

- To do so through group policy, go to **Computer Configuration\Administrative Templates\Windows Components\Windows Update** and select the setting to **Turn off auto-restart notifications for update installations**. This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

- To do so through MDM, use SetAutoRestartNotificationDisable. This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Scheduled automatic restart warnings

Since users aren't able to postpone a scheduled restart once the deadline is reached, you can configure a warning reminder before the scheduled restart. You can also configure a warning before the restart, to notify users once the restart is imminent and allow them to save their work.

To configure both through group policy, find the setting to **Configure auto-restart warning notifications schedule for updates** under **Computer Configuration\Administrative Templates\Windows Components\Windows Update**. The warning reminder can be configured by **Reminder (hours)** and the warning before an imminent automatic restart can be configured by **Warning (mins)**. This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

In MDM, to configure the warning reminder, use ScheduleRestartWarning. This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release. To configure the automatic restart imminent warning, use ScheduleImminentRestartWarning. This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Engaged restart

Engaged restart is the period of time when users are required to schedule a restart. Initially, Windows auto-restarts outside of working hours. Once the default seven day period ends, Windows transitions to user scheduled restarts.

You can adjust the following settings for engaged restart:

- Period of time before automatic restart transitions to engaged restart.
- The number of days that users can snooze engaged restart reminder notifications.
- The number of days before a pending restart automatically executes outside of working hours.

In group policy, go to **Computer Configuration\Administrative Templates\Windows Components\Windows Update** and use the setting to **Specify engaged restart transition and notification schedule for updates**. This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

In MDM, use the following policies:

- EngagedRestartTransitionSchedule
  - This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.
- EngagedRestartSnoozeSchedule
  - This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.
- EngagedRestartDeadline
  - This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Group policy settings for restart

In the group policy editor, the policy settings for restart behavior are in **Computer Configuration\Administrative Templates\Windows Components\Windows Update**. The following table shows which policies apply to Windows 10.

Expand table

Policy	Applies to Windows 10	Notes
Turn off auto-restart for updates during active hours	Yes	Use this policy to configure active hours, during which the device won't restart. This policy has no effect if the **No auto-restart with logged on users for scheduled automatic updates installations** or **Always automatically restart at the scheduled time** policies are enabled.
Always automatically restart at the scheduled time	Yes	Use this policy to configure a restart timer (between 15 and 180 minutes) that will start immediately after Windows Update installs important updates. This policy has no effect if the **No auto-restart with logged on users for scheduled automatic updates installations** policy is enabled. This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.
Specify deadline before auto-restart for update installation	Yes	Use this policy to specify how many days (between 2 and 14) an automatic restart can be delayed. This policy has no effect if the **No auto-restart with logged on users for scheduled automatic updates installations** or **Always automatically restart at the scheduled time** policies are enabled. This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.
No auto-restart with logged on users for scheduled automatic updates installations	Yes	Use this policy to prevent automatic restart when a user is logged on. This policy applies only when you configure the policy to **Configure Automatic Updates** to schedule the installation. The **No auto-restart with logged on users for scheduled automatic updates installation** policy was never created as a CSP. In Group Policy this policy doesn't work exactly as per description. This policy can result in no quality update reboots period, given many users never log off. The recommendation to replace this would be to leverage compliance deadline and then to configure no-auto reboot to prevent non-user aware reboots prior to the deadline being reached. For server devices, leverage Configure Automatic Updates options 7 - notify to install and notify to reboot.
Re-prompt for restart with scheduled installations	No	This policy isn't implemented on Windows 10 and has no effect if set on Windows 10 or Windows 11.
Delay Restart for scheduled installations	No	This policy isn't implemented on Windows 10 and has no effect if set on Windows 10 or Windows 11.
Reschedule Automatic Updates	No	This policy isn't implemented on Windows 10 and has no effect if set on Windows 10 or Windows 11.

Policy	Applies to Windows 10	Notes
scheduled installations		

> **Note**
>
> - You can only choose one path for restart behavior.
> - If you set conflicting restart policies, the actual restart behavior may not be what you expected.
> - When using RDP, only active RDP sessions are considered as signed-in users.
> - The **No auto-restart with logged on users for scheduled automatic updates installation** policy was never created as a CSP. In Group Policy this policy doesn't work exactly as per description. This policy can result in no quality update reboots period, given many users never log off. The recommendation to replace this would be to leverage compliance deadline and then to configure no-auto reboot to prevent non-user aware reboots prior to the deadline being reached. For server devices, leverage Configure Automatic Updates options 7 - notify to install and notify to reboot.

Registry keys used to manage restart

The following tables list registry values that correspond to the group policy settings for controlling restarts after updates in Windows 10.

`HKLM\Software\Policies\Microsoft\Windows\WindowsUpdate`

Expand table

Registry key	Key type	Value
`ActiveHoursEnd`	`REG_DWORD`	`0-23`: Set active hours to end at a specific hour. It starts with 12 AM (`0`) and ends with 11 PM (`23`).
`ActiveHoursStart`	`REG_DWORD`	`0-23`: Set active hours to start at a specific hour. It starts with 12 AM (`0`) and ends with 11 PM (`23`.)
`SetActiveHours`	`REG_DWORD`	`0`: Disable automatic restart after updates outside of active hours.

Registry key	Key type	Value
		`1`: Enable automatic restart after updates outside of active hours.

`HKLM\Software\Policies\Microsoft\Windows\WindowsUpdate\AU`

Expand table

Registry key	Key type	Value
`AlwaysAutoRebootAtScheduledTime`	`REG_DWORD`	`0`: Disable automatic restart after update installation at the scheduled time. `1`: Enable automatic restart after update installation at a scheduled time.
`AlwaysAutoRebootAtScheduledTimeMinutes`	`REG_DWORD`	`15-180`: Set automatic restart to occur after the specified number of minutes.
`AUOptions`	`REG_DWORD`	`2`: Notify for download and notify for installation of updates. `3`: Automatically download and notify for installation of updates. `4`: Automatically download and schedule installation of updates. `5`: Allow the local administrator to configure these settings. **Note:** To configure restart behavior, set this value to `4`.
`NoAutoRebootWithLoggedOnUsers`	`REG_DWORD`	`0`: If users are signed in, automatically restart ("disable don't reboot"). `1`: If a user is signed in, don't restart after an update installation. **Note:** If disabled (`0`), Automatic Updates notifies the user that the computer is scheduled to automatically restart in five minutes to complete the installation.
`ScheduledInstallTime`	`REG_DWORD`	`0-23`: Schedule update installation time to a specific hour. It starts with 12 AM (`0`) and ends with 11 PM (`23`).

There are three different registry combinations for controlling restart behavior:

- To set active hours:
  - `SetActiveHours` should be `1`.

  - Then to define the time range, use `ActiveHoursStart` and `ActiveHoursEnd`.

- To schedule a specific installation and restart time:
  - `AUOptions` should be `4`.
  - `ScheduledInstallTime` should specify the installation time.
  - Set `AlwaysAutoRebootAtScheduledTime` to `1`. This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.
  - `AlwaysAutoRebootAtScheduledTimeMinutes` should specify the number of minutes to wait before restarting. This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

- To delay restarting if a user is signed in:
  - `AUOptions` should be `4`.
  - Set `NoAutoRebootWithLoggedOnUsers` to `1`.

More resources

- Overview of Windows as a service
- Configure Delivery Optimization for Windows updates
- Configure Windows Update client policies
- Walkthrough: use group policy to configure Windows Update client policies
- Manage Windows software updates in Microsoft Intune

Last updated on 09/30/2025

Manage additional Windows Update settings

Article • 12/27/2024 • Applies to: ✅ Windows 11, ✅ Windows 10

> **Looking for consumer information?** See Windows Update: FAQ

You can use Group Policy settings or mobile device management (MDM) to configure the behavior of Windows Update on your Windows 10 devices. You can configure the update detection frequency, select when updates are received, specify the update service location and more.

Summary of Windows Update settings

Expand table

Group Policy setting	MDM setting	Supported from version
Specify Intranet Microsoft update service location	UpdateServiceUrl and UpdateServiceUrlAlternate	All
Automatic Updates Detection Frequency	DetectionFrequency	1703
Remove access to use all Windows Update features	Update/SetDisableUXWUAccess	All
Do not connect to any Windows Update Internet locations		All
Enable client-side targeting		All
Allow signed updates from an intranet Microsoft update service location	AllowNonMicrosoftSignedUpdate	All
Do not include drivers with Windows Updates	ExcludeWUDriversInQualityUpdate	1607

Group Policy setting	MDM setting	Supported from version
Configure Automatic Updates	AllowAutoUpdate	All
	Windows Update notifications display organization name *Organization name is displayed by default. A registry value can disable this behavior.	Windows 11 devices that are Microsoft Entra joined or registered
	Allow Windows updates to install before initial user sign-in (registry only)	Windows 11 version 22H2 with 2023-04 Cumulative Update Preview, or a later cumulative update

> **Important**
>
> Additional information about settings to manage device restarts and restart notifications for updates is available on **Manage device restarts after updates**.
>
> Additional settings that configure when feature and quality updates are received are detailed on **Configure Windows Update client policies**.

Scanning for updates

Admins have flexibility in configuring how their devices scan and receive updates.

Specify Intranet Microsoft update service location allows admins to point devices to an internal Microsoft update service location, while Do not connect to any Windows Update Internet locations gives them the option to restrict devices to just that internal update service. Automatic Updates Detection Frequency controls how frequently devices scan for updates.

You can make custom device groups that work with your internal Microsoft update service by using Enable client-side targeting. You can also make sure your devices receive updates that weren't signed by Microsoft from your internal Microsoft update service, through Allow signed updates from an intranet Microsoft update service location.

Finally, to make sure the updating experience is fully admin controlled, you can Remove access to use all Windows Update features for users.

For additional settings that configure when feature and quality updates are received, see Configure Windows Update client policies.

Specify intranet Microsoft update service location

Specifies an intranet server to host updates from Microsoft Update. You can then use this update service to automatically update computers on your network. This setting lets you specify a server on your network to function as an internal update service. The Automatic Updates client searches this service for updates that apply to the computers on your network.

To use this setting in Group Policy, go to **Computer Configuration\Administrative Templates\Windows Components\Windows Update\Specify Intranet Microsoft update service location**. You must set two server name values:

- The server from which the Automatic Updates client detects and downloads updates
- The server to which updated workstations upload statistics You can set both values to be the same server. An optional server name value can be specified to configure Windows Update Agent to download updates from an alternate download server instead of the intranet update service.

If the setting is set to **Enabled**, the Automatic Updates client connects to the specified intranet Microsoft update service (or alternate download server), instead of Windows Update, to search for and download updates. Enabling this setting means that end users in your organization don't have to go through a firewall to get updates, and it gives you the opportunity to test updates after deploying them. If the setting is set to **Disabled** or **Not Configured**, and if Automatic Updates isn't disabled by policy or user preference, the Automatic Updates client connects directly to the Windows Update site on the Internet.

The alternate download server configures the Windows Update Agent to download files from an alternative download server instead of the intranet update service. The option to download files with missing URLs allows content to be downloaded from the Alternate Download Server when there are no download URLs for files in the update metadata. This option should only be used when the intranet update service doesn't provide download URLs in the update metadata for files that are present on the alternate download server.

> **Note**

If the "Configure Automatic Updates" policy is disabled, then this policy has no effect.

If the "Alternate Download Server" is not set, it will use the intranet update service by default to download updates.

The option to "Download files with no Url..." is only used if the "Alternate Download Server" is set.

To configure this policy with MDM, use UpdateServiceUrl and UpdateServiceUrlAlternate.

Automatic Updates detection frequency

Specifies the hours that Windows will use to determine how long to wait before checking for available updates. The exact wait time is determined by using the hours specified here minus zero to twenty percent of the hours specified. For example, if this policy is used to specify a 20-hour detection frequency, then all clients to which this policy is applied will check for updates anywhere between 16 to 20 hours.

To set this setting with Group Policy, navigate to **Computer Configuration\Administrative Templates\Windows Components\Windows Update\Automatic Updates detection frequency**.

If the setting is set to **Enabled**, Windows checks for available updates at the specified interval. If the setting is set to **Disabled** or **Not Configured**, Windows checks for available updates at the default interval of 22 hours.

> **Note**
>
> The "Specify intranet Microsoft update service location" setting must be enabled for this policy to have effect.
>
> If the "Configure Automatic Updates" policy is disabled, this policy has no effect.

To configure this policy with MDM, use DetectionFrequency.

Remove access to use all Windows Update features

By enabling the Group Policy setting under **Computer Configuration\Administrative Templates\Windows Components\Windows update\Remove access to use all Windows update features**, administrators can disable the "Check for updates" option

for users. Any background update scans, downloads, and installations will continue to work as configured.

Do not connect to any Windows Update Internet locations

Even when Windows Update is configured to receive updates from an intranet update service, it will periodically retrieve information from the public Windows Update service to enable future connections to Windows Update, and other services like Microsoft Update or the Microsoft Store.

Use **Computer Configuration\Administrative Templates\Windows Components\Windows update\Do not connect to any Windows Update Internet locations** to enable this policy. When enabled, this policy will disable the functionality described above, and may cause connection to public services such as the Microsoft Store, Windows Update client policies, and Delivery Optimization to stop working.

> ⓘ **Note**
>
> This policy applies only when the device is configured to connect to an intranet update service using the "Specify intranet Microsoft update service location" policy.

Enable client-side targeting

Specifies the target group name or names that should be used to receive updates from an intranet Microsoft update service. This allows admins to configure device groups that receive different updates from sources like WSUS or Configuration Manager.

This Group Policy setting can be found under **Computer Configuration\Administrative Templates\Windows Components\Windows update\Enable client-side targeting**. If the setting is set to **Enabled**, the specified target group information is sent to the intranet Microsoft update service, which uses it to determine which updates should be deployed to this computer. If the setting is set to **Disabled** or **Not Configured**, no target group information is sent to the intranet Microsoft update service.

If the intranet Microsoft update service supports multiple target groups, this policy can specify multiple group names separated by semicolons. Otherwise, a single group must be specified.

> ⓘ **Note**

This policy applies only when the intranet Microsoft update service the device is directed to is configured to support client-side targeting. If the "Specify intranet Microsoft update service location" policy is disabled or not configured, this policy has no effect.

Allow signed updates from an intranet Microsoft update service location

This policy setting allows you to manage whether Automatic Updates accepts updates signed by entities other than Microsoft when the update is found on an intranet Microsoft update service location.

To configure this setting in Group Policy, go to **Computer Configuration\Administrative Templates\Windows Components\Windows update\Allow signed updates from an intranet Microsoft update service location**.

If you enable this policy setting, Automatic Updates accepts updates received through an intranet Microsoft update service location, as specified by Specify Intranet Microsoft update service location, if they're signed by a certificate found in the "Trusted Publishers" certificate store of the local computer. If you disable or don't configure this policy setting, updates from an intranet Microsoft update service location must be signed by Microsoft.

> **Note**
>
> Updates from a service other than an intranet Microsoft update service must always be signed by Microsoft and are not affected by this policy setting.

To configure this policy with MDM, use AllowNonMicrosoftSignedUpdate.

Installing updates

To add more flexibility to the update process, settings are available to control update installation.

Configure Automatic Updates offers four different options for automatic update installation, while Do not include drivers with Windows Updates makes sure drivers aren't installed with the rest of the received updates.

Do not include drivers with Windows Updates

Allows admins to exclude Windows Update drivers during updates.

To configure this setting in Group Policy, use **Computer Configuration\Administrative Templates\Windows Components\Windows update\Do not include drivers with Windows Updates**. Enable this policy to not include drivers with Windows quality updates. If you disable or don't configure this policy, Windows Update includes updates that have a Driver classification.

Configure Automatic Updates

Enables the IT admin to manage automatic update behavior to scan, download, and install updates.

Configuring Automatic Updates by using Group Policy

Under **Computer Configuration\Administrative Templates\Windows Components\Windows update\Configure Automatic Updates**, you must select one of the following options:

2 - Notify for download and auto install - When Windows finds updates that apply to this device, users are notified that updates are ready to be downloaded. After going to **Settings** > **Update & security** > **Windows Update**, users can download and install any available updates.

3 - Auto download and notify for Install - Windows finds updates that apply to the device and downloads them in the background (the user isn't notified or interrupted during this process). When the downloads are complete, users will be notified that they're ready to install. After going to **Settings** > **Update & security** > **Windows Update**, users can install them.

4 - Auto download and schedule the install - Specify the schedule using the options in the Group Policy Setting. For more information about this setting, see Schedule update installation.

5 - Allow local admin to choose setting - With this option, local administrators are allowed to use the settings app to select a configuration option of their choice. Local administrators won't be allowed to disable the configuration for Automatic Updates. This option isn't available in any Windows 10 or later versions.

7 - Notify for install and notify for restart (Windows Server 2016 and later only) - With this option, when Windows finds updates that apply to this device, they are downloaded, then users are notified that updates are ready to be installed. Once updates are installed, a notification is displayed to users to restart the device.

If this setting is set to **Disabled**, any updates that are available on Windows Update must be downloaded and installed manually. To do this, users must go to **Settings** > **Update & security** > **Windows Update**.

If this setting is set to **Not Configured**, an administrator can still configure Automatic Updates through the settings app, under **Settings** > **Update & security** > **Windows Update** > **Advanced options**.

Configuring Automatic Updates by editing the registry

> ⓘ **Note**
>
> Serious problems might occur if you modify the registry incorrectly by using Registry Editor or by using another method. These problems might require you to reinstall the operating system. Microsoft cannot guarantee that these problems can be resolved. Modify the registry at your own risk.

In an environment that doesn't have Active Directory deployed, you can edit registry settings to configure group policies for Automatic Update.

To do this, follow these steps:

1. Select **Start**, search for "regedit", and then open Registry Editor.
2. Open the following registry key:

   ```
   HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\WindowsUpdate\AU
   ```

3. Add one of the following registry values to configure Automatic Update.
   - NoAutoUpdate (REG_DWORD):
     - **0**: Automatic Updates is enabled (default).
     - **1**: Automatic Updates is disabled.
   - AUOptions (REG_DWORD):
     - **1**: Keep my computer up to date is disabled in Automatic Updates.
     - **2**: Notify of download and installation.

- **3**: Automatically download and notify of installation.
- **4**: Automatically download and scheduled installation.
- **5**: Allow local admin to select the configuration mode. This option isn't available for Windows 10 or later versions.
- **7**: Notify for install and notify for restart. (Windows Server 2016 and later only)

- ScheduledInstallEveryWeek (REG_DWORD):
  - **0**: Do not enforce a once-per-week scheduled installation
  - **1**: Enforce automatic installations once a week on the specified day and time. (Requires ***ScheduledInstallDay*** and ***ScheduledInstallTime*** to be set.)
- ScheduledInstallDay (REG_DWORD):
  - **0**: Every day.
  - **1** through **7**: The days of the week from Sunday (1) to Saturday (7).
- ScheduledInstallTime (REG_DWORD):

  n, where **n** equals the time of day in a 24-hour format (0-23).
- UseWUServer (REG_DWORD)

  Set this value to **1** to configure Automatic Updates to use a server that is running Software Update Services instead of Windows Update.
- RescheduleWaitTime (REG_DWORD)

  m, where **m** equals the time period to wait between the time Automatic Updates starts and the time that it begins installations where the scheduled times have passed. The time is set in minutes from 1 to 60, representing 1 minute to 60 minutes)

  > ⓘ **Note**
  >
  > This setting only affects client behavior after the clients have updated to the SUS SP1 client version or later versions.

- NoAutoRebootWithLoggedOnUsers (REG_DWORD):

0 (false) or **1** (true). If set to **1**, Automatic Updates doesn't automatically restart a computer while users are logged on.

> ⓘ **Note**
>
> This setting affects client behavior after the clients have updated to the SUS SP1 client version or later versions.

To use Automatic Updates with a server that is running Windows Software Update Services (WSUS), see the Deploying Microsoft Windows Server Update Services guidance.

When you configure Automatic Updates directly by using the policy registry keys, the policy overrides the preferences that are set by the local administrative user to configure the client. If an administrator removes the registry keys at a later date, the preferences that were set by the local administrative user are used again.

To determine the WSUS server that the client computers and servers connect to for updates, add the following registry values to the registry:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\WindowsUpdate\
```

- WUServer (REG_SZ)

  This value sets the WSUS server by HTTP name (for example, http://IntranetSUS).

- WUStatusServer (REG_SZ)

  This value sets the SUS statistics server by HTTP name (for example, http://IntranetSUS).

Display organization name in Windows Update notifications

When Windows 11 clients are associated with a Microsoft Entra tenant, the organization name appears in the Windows Update notifications. For instance, when you have a compliance deadline configured for Windows Update client policies, the user notification displays a message similar to **Contoso requires important updates to be installed**. The organization name will also display on the **Windows Update** page in the **Settings** for Windows 11.

The organization name appears automatically for Windows 11 clients that are associated with Microsoft Entra ID in any of the following ways:

- Microsoft Entra joined
- Microsoft Entra registered
- Microsoft Entra hybrid joined

To disable displaying the organization name in Windows Update notifications, add or modify the following values in the registry:

- **Registry key**: `HKEY_LOCAL_MACHINE\Software\Microsoft\WindowsUpdate\Orchestrator\Configurations`
- **DWORD value name**: UsoDisableAADJAttribution
- **Value data:** 1

The following PowerShell script is provided as an example to you:

PowerShell

```
$registryPath =
"HKLM:\Software\Microsoft\WindowsUpdate\Orchestrator\Configurations"
$Name = "UsoDisableAADJAttribution"
$value = "1"

if (!(Test-Path $registryPath))
{
  New-Item -Path $registryPath -Force | Out-Null
}

New-ItemProperty -Path $registryPath -Name $name -Value $value -PropertyType
DWORD -Force | Out-Null
```

Allow Windows updates to install before initial user sign-in

(Starting in Windows 11, version 22H2 with 2023-04 Cumulative Update Preview, or a later cumulative update)

On new devices, Windows Update doesn't begin installing background updates until a user has completed the Out of Box Experience (OOBE) and signs in for the first time. In many cases, the user signs in immediately after completing the OOBE. However, some VM-based solutions provision a device and automate the first user experience. These

VMs may not be immediately assigned to a user so they won't see an initial sign-in until several days later.

In scenarios where initial sign-in is delayed, setting the following registry values allow devices to begin background update work before a user first signs in:

- **Registry key**: `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\WindowsUpdate\Orchestrator`
- **DWORD value name**: ScanBeforeInitialLogonAllowed
- **Value data**: 1

> ⚠ **Warning**
>
> This value is designed to be used only for scenarios with a deferred initial user sign in. Setting this value on devices where initial user sign in isn't delayed could have a detrimental effect on performance since it may allow update work to occur as the user is signing in for the first time.

Feedback

Was this page helpful? Yes No

Provide product feedback

What are Windows Update client policies?

Article • 03/31/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

> **Looking for consumer information?** See Windows Update: FAQ.

Windows Update client policies are a free service that is available for the following editions of Windows 10 and Windows 11:

- Pro, including Pro for Workstations
- Education
- Enterprise, including Enterprise LTSC, IoT Enterprise, and IoT Enterprise LTSC

Windows Update client policies enable IT administrators to keep their organization's Windows client devices always up to date with the latest security updates and Windows features by directly connecting these systems to the Windows Update service. You can use Group Policy or Mobile Device Management (MDM) solutions, such as Microsoft Intune, to configure the Windows Update client policies that control how and when devices are updated.

> **Tip**
>
> This feature was formerly known as *Windows Update for Business*.

Specifically, Windows Update client policies let you control update offerings and experiences to allow for reliability and performance testing on a subset of devices before deploying updates across the organization. It also provides a positive update experience for people in your organization.

What can I do with Windows Update client policies?

Windows Update client policies enable commercial customers to manage which Windows Updates are received along with the experience a device has when it receives them.

You can control Windows Update client policies by using either MDM tools or Group Policy management, such as local group policy or the Group Policy Management Console (GPMC), and various other non-Microsoft management tools. MDMs use

Configuration Service Provider (CSP) policies instead of Group Policy. Intune additionally uses Cloud Policies. Not all policies are available in all formats (CSP, Group Policy, or Cloud Policy).

Manage deployment of Windows Updates

By using Windows Update client policies, you can:

- Control the types of Windows Updates are offered to devices in your organization
- Control when updates are applied to the devices
- Deploy updates to devices in your organization in waves

Manage which updates are offered

Windows Update client policies enable an IT administrator to receive and manage various types of Windows Updates.

Types of updates managed by Windows Update client policies

Windows Update client policies provide management policies for several types of updates to Windows 10 devices:

- **Feature updates:** Previously referred to as upgrades, feature updates contain not only security and quality revisions, but also significant feature additions and changes. Feature updates are released as soon as they become available. Feature updates aren't available for LTSC devices.
- **Quality updates:** Quality updates are traditional operating system updates. Typically quality updates are released on the second Tuesday of each month, though they can be released at any time. These include security, critical, and driver updates.
- **Driver updates:** Updates for non-Microsoft drivers that are relevant to your devices. Driver updates are on by default, but you can use Windows Update client policies to turn them off if you prefer.
- **Microsoft product updates**: Updates for other Microsoft products, such as versions of Office that are installed by using Windows Installer (MSI). Versions of Office that are installed by using Click-to-Run can't be updated by using Windows Update client policies. Product updates are off by default. You can turn them on by using Windows Update client policies.

Offering

You can control when updates are applied. For example, you can defer when an update is installed on a device or by pausing updates for a certain period.

Manage when updates are offered

You can defer or pause the installation of updates for a set period of time.

Enroll in prerelease updates

The branch readiness level enables administrators to specify which channel of feature updates they want to receive. Today there are branch readiness level options for both prerelease and released updates:

- Windows Insider Canary channel
- Windows Insider Dev channel
- Windows Insider Beta channel
- Windows Insider Release Preview channel

Defer an update

An administrator can defer the installation of both feature and quality updates from deploying to devices within a range of time based on when those updates are first made available on the Windows Update service. You can use this deferral to allow time to validate deployments as they're pushed to devices. Deferrals work by allowing you to specify the number of days after an update is released before it's offered to a device. That is, if you set a feature update deferral period of 365 days, the device won't install a feature update that has been released for less than 365 days. To defer feature updates, use the **Select when Preview Builds and feature updates are Received** policy.

Expand table

Category	Maximum deferral period
Feature updates	365 days
Quality updates	30 days
Nondeferrable	none

Pause an update

If you discover a problem while deploying a feature or quality update, you can pause the update for 35 days from a specified start date to prevent other devices from installing it until the issue is mitigated. If you pause a feature update, quality updates are still offered to devices to ensure they stay secure. The pause period for both feature and quality updates is calculated from a start date that you set.

To pause feature updates, use the **Select when Preview Builds and feature updates are Received** policy and to pause quality updates use the **Select when Quality Updates are Received** policy. For more information, see Pause feature updates and Pause quality updates.

Built-in benefits: When updating from Windows Update, you get the added benefits of built-in compatibility checks to prevent against a poor update experience for your device and a check to prevent repeated rollbacks.

Recommendations

For the best experience with Windows Update, follow these guidelines:

- Use devices for at least 6 hours per month, including at least 2 hours of continuous use.
- Keep devices regularly charged. Plugging in devices overnight enables them to automatically update outside of active hours.
- Make sure that devices have at least 10 GB of free space.
- Give devices unobstructed access to the Windows Update service.

Manage the end-user experience when receiving Windows Updates

Windows Update client policies provide controls to help meet your organization's security standards and provide a great end-user experience. We do this by enabling you to set automatic updates at times that work well for people in your organization and set deadlines for quality and feature updates. Because Windows Update includes built-in intelligence, it's better to use fewer controls to manage the user experience.

Recommended experience settings

Features like active hours help provide the best experience for end users while keeping devices more secure and up to date. Follow these steps to take advantage of these features:

1. Automatically download, install, and restart (default if no restart policies are set up or enabled).
2. Use the default notifications.
3. Set update deadlines.

Setting deadlines

A compliance deadline policy enables you to set separate deadlines and grace periods for feature and quality updates.

This policy enables you to specify the number of days from an update's publication date that it must be installed on the device. The policy also includes a configurable grace period that specifies the number of days from when the update is installed on the device until the device is forced to restart. This approach is useful in a vacation scenario as it allows, for example, users who have been away to have a bit of time before being forced to restart their devices when they return from vacation.

Update Baseline

> **Note**
>
> The Update Baseline toolkit is available only for Group Policy. Update Baseline does not affect your offering policies, whether you're using deferrals or target version to manage which updates are offered to your devices when. Update Baseline is not currently supported for Windows 11.

The large number of different policies offered can be overwhelming. Update Baseline provides a clear list of recommended Windows update policy settings for IT administrators who want the best user experience while also meeting their update compliance goals. The Update Baseline for Windows 10 includes policy settings recommendations covering deadline configuration, restart behavior, power policies, and more.

The Update Baseline toolkit makes it easy by providing a single command for IT Admins to apply the Update Baseline to devices. You can get the Update Baseline toolkit from the Download Center.

Other Windows Update client policies services

The following services are part of the Windows Update product family:

- Windows Update for Business reports is a cloud-based solution that provides information about your Microsoft Entra joined devices' compliance with Windows updates. Windows Update for Business reports is offered through the Azure portal. Windows Update for Business reports helps you:
  - Monitor security, quality, driver, and feature updates for Windows 11 and Windows 10 devices
  - Report on devices with update compliance issues
  - Analyze and display your data in multiple ways

- Windows Autopatch is a cloud service designed to work with your existing Windows Update client policies. Windows Autopatch provides additional control over the approval, scheduling, and safeguarding of updates delivered from Windows Update to managed devices.

Feedback

Was this page helpful? Yes No

Provide product feedback

Configure Windows Update client policies

Applies to: ✅ Windows 11, ✅ Windows 10, ✅ Windows Server 2022, ✅ Windows Server 2019, ✅ Windows Server 2016

> **Looking for consumer information?** See Windows Update: FAQ.

You can use Group Policy or your mobile device management (MDM) service to configure Windows Update client policies for your devices. The sections in this article provide the Group Policy and MDM policies for Windows 10, version 1511 and later, including Windows 11. The MDM policies use the OMA-URI setting from the Policy CSP.

> **Note**
>
> Windows Server *doesn't* get feature updates from Windows Update, so only the quality update policies apply. This behavior doesn't apply to Azure Stack hyperconverged infrastructure (HCI).

You can use Windows Update client policies regardless of the diagnostic data level that you chose. If the diagnostic data level is set to **0 (Security)**, Windows Update client policies are still honored. For more information, see Configure the operating system diagnostic data level.

> **Tip**
>
> This feature was formerly known as *Windows Update for Business*.

Start by grouping devices

By grouping devices with similar deferral periods, administrators are able to cluster devices into deployment or validation groups, which can be as a quality control measure as updates are deployed. With deferral windows and the ability to pause updates, administrators can effectively control and measure update deployments, updating a small pool of devices first to verify quality, prior to a broader roll-out to their organization.

> **Tip**
>
> In addition to setting up multiple rings for your update deployments, also incorporate devices enrolled in the Windows Insider Program as part of your deployment strategy. This will provide you the chance to not only evaluate new features before they are broadly available to the public, but it also increases the lead time to provide feedback and influence Microsoft's design on functional aspects of the product. For more information on Windows Insider program, see https://insider.windows.com/.

Configure devices for the appropriate service channel

With Windows Update client policies, you can set a device to be on either Windows Insider Preview or the General Availability Channel servicing branch. For more information on this servicing model, see Servicing channels.

Release branch policies

Expand table

Policy	Sets registry key under HKLM\Software
GPO for Windows 10, version 1607 or later: Computer Configuration > Administrative Templates > Windows Components > Windows Update > Windows Update for Business > **Select when feature updates are received** This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.	\Policies\Microsoft\Windows\WindowsUpdate\BranchReadinessLevel
GPO for Windows 10, version 1511: Computer Configuration > Administrative Templates > Windows Components > Windows Update > **Defer Upgrades and Updates** This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.	\Policies\Microsoft\Windows\WindowsUpdate\DeferUpgrade
MDM for Windows 10, version 1607 or later: ../Vendor/MSFT/Policy/Config/Update/**BranchReadinessLevel** This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.	\Microsoft\PolicyManager\default\Update\BranchReadinessLevel
MDM for Windows 10, version 1511: ../Vendor/MSFT/Policy/Config/Update/**RequireDeferUpgrade** This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.	\Microsoft\PolicyManager\default\Update\RequireDeferUpgrade

Starting with Windows 10, version 1703, users can configure the branch readiness level for their device by using **Settings** > **Update & security** > **Windows Update** > **Advanced options**.

> **Note**
>
> Users will not be able to change this setting if it was configured by policy.

Configure when devices receive feature updates

After you configure the servicing branch (Windows Insider Preview or General Availability Channel), you can then define if, and for how long, you would like to defer receiving feature updates following their availability from Microsoft on Windows Update. You can defer receiving these feature updates for a period of up to 365 days from their release by setting the `DeferFeatureUpdatesPeriodinDays` value.

For example, a device on the General Availability Channel with `DeferFeatureUpdatesPeriodinDays=30` won't install a feature update that is first publicly available on Windows Update in September until 30 days later, in October.

Policy settings for deferring feature updates

Expand table

Policy	Sets registry key under HKLM\Software
GPO for Windows 10, version 1607 or later: Computer Configuration > Administrative Templates > Windows Components > Windows Update > Windows Update for Business > **Select when feature updates are received** This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.	\Policies\Microsoft\Windows\WindowsUpdate\DeferFeatureUpdates \Policies\Microsoft\Windows\WindowsUpdate\DeferFeatureUpdatesPeriodInDays
GPO for Windows 10, version 1511: Computer Configuration > Administrative Templates > Windows Components > Windows Update > **Defer Upgrades and Updates** This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.	\Policies\Microsoft\Windows\WindowsUpdate\DeferUpgradePeriod
MDM for Windows 10, version 1607 and later: ../Vendor/MSFT/Policy/Config/Update/ **DeferFeatureUpdatesPeriodInDays**	\Microsoft\PolicyManager\default\Update\DeferFeatureUpdatesPeriodInDays
MDM for Windows 10, version 1511: ../Vendor/MSFT/Policy/Config/Update/ **DeferUpgrade** This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.	\Microsoft\PolicyManager\default\Update\RequireDeferUpgrade

> **Note**
>
> If not configured by policy, individual users can defer feature updates by using **Settings** > **Update & security** > **Windows Update** > **Advanced options**.

Pause feature updates

You can also pause a device from receiving feature updates by a period of up to 35 days from when the value is set. After 35 days have passed, the pause setting will automatically expire and the device will scan Windows Update for applicable feature updates. Following this scan, you can then pause feature updates for the device again.

Starting with Windows 10, version 1703, when you configure a pause by using policy, you must set a start date for the pause to begin. The pause period is calculated by adding 35 days to this start date.

In cases where the pause policy is first applied after the configured start date has passed, you can extend the pause period up to a total of 35 days by configuring a later start date.

> **Important**
>
> In Windows 10, version 1703 and later versions, you can pause feature updates to 35 days, similar to the number of days for quality updates.

Policy settings for pausing feature updates

Expand table

Policy	Sets registry key under HKLM\Software
GPO for Windows 10, version 1607 or later: Computer Configuration > Administrative Templates > Windows Components > Windows Update > Windows Update for Business > **Select when feature updates are received**	**1607:** \Policies\Microsoft\Windows\WindowsUpdate\PauseFeatureUpdates **1703 and later:** \Policies\Microsoft\Windows\WindowsUpdate\PauseFeatureUpdatesStartTime
GPO for Windows 10, version 1511: Computer Configuration > Administrative Templates > Windows Components > Windows Update > **Defer Upgrades and Updates** This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.	\Policies\Microsoft\Windows\WindowsUpdate\Pause
MDM for Windows 10, version 1607 or later: ../Vendor/MSFT/Policy/Config/Update/**PauseFeatureUpdates**	**1607:** \Microsoft\PolicyManager\default\Update\PauseFeatureUpdates **1703 and later:** \Microsoft\PolicyManager\default\Update\PauseFeatureUpdatesStartTime
MDM for Windows 10, version 1511: ../Vendor/MSFT/Policy/Config/Update/**DeferUpgrade** This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.	\Microsoft\PolicyManager\default\Update\Pause

You can check the date that feature updates were paused by checking the registry key **PausedFeatureDate** under **HKLM\SOFTWARE\Microsoft\WindowsUpdate\UpdatePolicy\Settings**.

The local group policy editor (GPEdit.msc) won't reflect whether the feature update pause period has expired. Although the device will resume feature updates after 35 days automatically, the pause check box will remain selected in the policy editor. To check whether a device has automatically resumed taking feature updates, check the status registry key **PausedFeatureStatus** under **HKLM\SOFTWARE\Microsoft\WindowsUpdate\UpdatePolicy\Settings** for the following values:

Expand table

Value	Status
0	feature updates not paused
1	feature updates paused
2	feature updates have auto-resumed after being paused

> **Note**
>
> If not configured by policy, individual users can pause feature updates by using **Settings** > **Update & security** > **Windows Update** > **Advanced options**.

Starting with Windows 10, version 1703, using Settings to control the pause behavior provides a more consistent experience, specifically:

- Any active restart notifications are cleared or closed.

- Any pending restarts are canceled.
- Any pending update installations are canceled.
- Any update installation running when pause is activated will attempt to roll back.

Configure when devices receive quality updates

Quality updates are typically published on the second Tuesday of every month, although they can be released at any time. You can define if, and for how long, you would like to defer receiving quality updates following their availability. You can defer receiving these quality updates for a period of up to 30 days from their release by setting the **DeferQualityUpdatesPeriodinDays** value.

You can set your system to receive updates for other Microsoft products—known as Microsoft updates (such as Microsoft Office, Visual Studio)—along with Windows updates by setting the **AllowMUUpdateService** policy. When you do this, these Microsoft updates will follow the same deferral and pause rules as all other quality updates. For a list of other Microsoft products that might be updated, see Update other Microsoft products.

Policy settings for deferring quality updates

Expand table

Policy	Sets registry key under HKLM\Software
GPO for Windows 10, version 1607 or later: Computer Configuration > Administrative Templates > Windows Components > Windows Update > Windows Update for Business > **Select when Quality Updates are received**	\Policies\Microsoft\Windows\WindowsUpdate\DeferQualityUpdates \Policies\Microsoft\Windows\WindowsUpdate\DeferQualityUpdatesPeriodInDays
GPO for Windows 10, version 1511: Computer Configuration > Administrative Templates > Windows Components > Windows Update > **Defer Upgrades and Updates** This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.	\Policies\Microsoft\Windows\WindowsUpdate\DeferUpdatePeriod
MDM for Windows 10, version 1607 or later: ../Vendor/MSFT/Policy/Config/Update/**DeferQualityUpdatesPeriodInDays**	\Microsoft\PolicyManager\default\Update\DeferQualityUpdatesPeriodInDays
MDM for Windows 10, version 1511: ../Vendor/MSFT/Policy/Config/Update/**DeferUpgrade** This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.	\Microsoft\PolicyManager\default\Update\RequireDeferUpdate

> **Note**
>
> If not configured by policy, individual users can defer quality updates by using **Settings** > **Update & security** > **Windows Update** > **Advanced options**.

Pause quality updates

You can also pause a system from receiving quality updates for a period of up to 35 days from when the value is set. After 35 days have passed, the pause setting will automatically expire and the device will scan Windows Update for applicable quality updates. Following this scan, you can then pause quality updates for the device again.

Starting with Windows 10, version 1703, when you configure a pause by using policy, you must set a start date for the pause to begin. The pause period is calculated by adding 35 days to this start date.

In cases where the pause policy is first applied after the configured start date has passed, you can extend the pause period up to a total of 35 days by configuring a later start date.

> **Note**
>
> Starting with Windows 10, version 1809, IT administrators can prevent individual users from pausing updates.

Policy settings for pausing quality updates

Expand table

Policy	Sets registry key under HKLM\Software
GPO for Windows 10, version 1607 or later: Computer Configuration > Administrative Templates > Windows Components > Windows Update > Windows Update for Business > **Select when Quality Updates are received**	**1607:** \Policies\Microsoft\Windows\WindowsUpdate\PauseQualityUpdates **1703:** \Policies\Microsoft\Windows\WindowsUpdate\PauseQualityUpdatesStartTime
GPO for Windows 10, version 1511: Computer Configuration > Administrative Templates > Windows Components > Windows Update > **Defer Upgrades and Updates** This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.	\Policies\Microsoft\Windows\WindowsUpdate\Pause
MDM for Windows 10, version 1607 or later: ../Vendor/MSFT/Policy/Config/Update/**PauseQualityUpdates**	**1607:** \Microsoft\PolicyManager\default\Update\PauseQualityUpdates **1703:** \Microsoft\PolicyManager\default\Update\PauseQualityUpdatesStartTime
MDM for Windows 10, version 1511: ../Vendor/MSFT/Policy/Config/Update/**DeferUpgrade** This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.	\Microsoft\PolicyManager\default\Update\Pause

You can check the date that quality updates were paused by checking the registry key **PausedQualityDate** under **HKLM\SOFTWARE\Microsoft\WindowsUpdate\UpdatePolicy\Settings**.

The local group policy editor (GPEdit.msc) won't reflect whether the quality update pause period has expired. Although the device will resume quality updates after 35 days automatically, the pause check box will remain selected in the policy editor. To check whether a device has automatically resumed taking quality Updates, check the status registry key **PausedQualityStatus** under **HKLM\SOFTWARE\Microsoft\WindowsUpdate\UpdatePolicy\Settings** for the following values:

Expand table

Value	Status
0	quality updates not paused
1	quality updates paused
2	quality updates have auto-resumed after being paused

> **Note**
>
> If not configured by policy, individual users can pause quality updates by using **Settings** > **Update & security** > **Windows Update** > **Advanced options**.

Starting with Windows 10, version 1703, using Settings to control the pause behavior provides a more consistent experience, specifically:

- Any active restart notifications are cleared or closed
- Any pending restarts are canceled
- Any pending update installations are canceled
- Any update installation running when pause is activated will attempt to roll back

Configure when devices receive Windows Insider Preview builds

Starting with Windows 10, version 1709, you can set policies to manage preview builds and their delivery:

The **Manage preview builds** setting gives administrators control over enabling or disabling preview build installation on a device. You can also decide to stop preview builds once the release is public.

- Group Policy: **Computer Configuration/Administrative Templates/Windows Components/Windows Update/Windows Update for Business** - *Manage preview builds*
- MDM: **Update/ManagePreviewBuilds**
- Microsoft Configuration Manager: **Enable dual scan, manage through Windows Update for Business policy**

> **Important**
>
> This policy replaces the "Toggle user control over Insider builds" policy under that is only supported up to Windows 10, version 1703. You can find the older policy here:
>
> - Group Policy: **Computer Configuration/Administrative Templates/Windows Components/Data Collection and Preview Builds/Toggle user control over Insider builds**
> - MDM: **System/AllowBuildPreview**

The policy settings to **Select when feature updates are received** allows you to choose between preview flight rings, and allows you to defer or pause their delivery.

- Group Policy: **Computer Configuration/Administrative Templates/Windows Components/Windows Update/Windows Update for Business** - *Select when Preview Builds and feature updates are received*
- MDM: **Update/BranchReadinessLevel**

Exclude drivers from quality updates

Starting with Windows 10, version 1607, you can selectively opt out of receiving driver update packages as part of your normal quality update cycle. This policy won't apply to updates to drivers provided with the operating system (which will be packaged within a security or critical update) or to feature updates, where drivers might be dynamically installed to ensure the feature update process can complete.

Policy settings to exclude drivers

Expand table

Policy	Sets registry key under HKLM\Software
GPO for Windows 10, version 1607 or later: Computer Configuration > Administrative Templates > Windows Components > Windows Update > **Do not include drivers with Windows Updates**	\Policies\Microsoft\Windows\WindowsUpdate\ExcludeWUDriversInQualityUpdate
MDM for Windows 10, version 1607 and later: ../Vendor/MSFT/Policy/Config/Update/**ExcludeWUDriversInQualityUpdate**	\Microsoft\PolicyManager\default\Update\ExcludeWUDriversInQualityUpdate

Enable optional updates

In addition to the monthly cumulative update, optional updates are available to provide new features and nonsecurity changes. Most optional updates are released on the fourth Tuesday of the month, known as optional nonsecurity preview releases. Optional updates can also include features that are gradually rolled out, known as controlled feature rollouts (CFRs). Installation of optional updates isn't enabled by default for devices that receive updates using Windows Update client policies. However, you can enable optional updates for devices by using the **Enable optional updates** policy.

To keep the timing of updates consistent, the **Enable optional updates** policy respects the deferral period for quality updates. This policy allows you to choose if devices should receive CFRs in addition to the optional nonsecurity preview releases, or if the end-user can make the decision to install optional updates. This policy can change the behavior of the **Get the latest updates as soon as they're available** option in **Settings** > **Update & security** > ***Windows Update** > **Advanced options**.

The following options are available for the policy:

- **Automatically receive optional updates (including CFRs)**:
  - The latest optional nonsecurity updates and CFRs are automatically installed on the device. The quality update deferral period is applied to the installation of these updates.
  - The **Get the latest updates as soon as they're available** option is selected and users can't change the setting.
  - Devices will receive CFRs in early phases of the rollout.

- **Automatically receive optional updates**:
  - The latest optional nonsecurity updates are automatically installed on the device but CFRs aren't. The quality update deferral period is applied to the installation of these updates.
  - The **Get the latest updates as soon as they're available** option isn't selected and users can't change the setting.

- **Users can select which optional updates to receive**:
  - Users can select which optional updates to install from **Settings** > **Update & security** > **Windows Update** > **Advanced options** > **Optional updates**.
    - Optional updates are offered to the device, but user interaction is required to install them unless the **Get the latest updates as soon as they're available** option is also enabled.
    - CFRs are offered to the device, but not necessarily in the early phases of the rollout.
  - Users can enable the **Get the latest updates as soon as they're available** option in **Settings** > **Update & security** > ***Windows Update** > **Advanced options**. If the user enables the **Get the latest updates as soon as they're available**, then:
    - The device will receive CFRs in early phases of the rollout.
    - Optional updates are automatically installed on the device.

- **Not configured** (default):

- Optional updates aren't installed on the device and the **Get the latest updates as soon as they're available** option is disabled.

Policies to enable optional updates

Expand table

Policy	Sets registry key under HKLM\Software
GPO applies to: • Windows 11, version 22H2 with KB5029351, and later versions • Windows 10, version 22H2 with KB5032278, or a later cumulative update installed **GPO location**: Computer Configuration > Administrative Templates > Windows Components > Windows Update > Manage updates offered from Windows Update > **Enable optional updates**	\Policies\Microsoft\Windows\WindowsUpdate\AllowOptionalContent
MDM applies to: • Windows 11, version 22H2 with KB5029351 and later versions • Windows 10, version 22H2 with KB5032278, or a later cumulative update installed **MDM location**: ./Device/Vendor/MSFT/Policy/Config/Update/AllowOptionalContent	\Policies\Microsoft\Windows\WindowsUpdate\AllowOptionalContent

Enable features that are behind temporary enterprise feature control

New features and enhancements are introduced through the monthly cumulative update to provide continuous innovation for Windows 11. To give organizations time to plan and prepare, some of these new features are temporarily turned off by default. Features that are turned off by default are listed in the KB article for the monthly cumulative update. Typically, a feature is selected to be off by default because it either impacts the user experience or IT administrators significantly.

The features that are behind temporary enterprise feature control will be enabled in the next annual feature update. Organizations can choose to deploy feature updates at their own pace, to delay these features until they're ready for them. For a list of features that are turned off by default, see Windows 11 features behind temporary enterprise control.

Policy settings to enable features that are behind temporary enterprise control

Expand table

Policy	Sets registry key under HKLM\Software
GPO for Windows 11, version 22H2 with KB5022845 and later: Computer Configuration > Administrative	\Policies\Microsoft\Windows\WindowsUpdate\AllowTemporaryEnterpriseFeatureControl

Policy	Sets registry key under HKLM\Software
Templates > Windows Components > Windows Update > Manage end user experience > **Enable features introduced via servicing that are off by default**	
MDM for Windows 11, version 22H2 with KB5022845 and later: ./Device/Vendor/MSFT/Policy/Config/Update/AllowTemporaryEnterpriseFeatureControl	\Microsoft\PolicyManager\default\Update\AllowTemporaryEnterpriseFeatureControl

Summary: MDM and Group Policy settings for Windows 10, version 1703 and later

The following are quick-reference tables of the supported policy values for Windows Update client policies in Windows 10, version 1607 and later.

GPO: HKLM\Software\Policies\Microsoft\Windows\WindowsUpdate

Expand table

GPO Key	Key type	Value
AllowOptionalContent *Added in*: • Windows 11, version 22H2 with KB5029351 and later • Windows 10, version 22H2 with KB5032278, or a later cumulative update installed	REG_DWORD	1: Automatically receive optional updates (including CFRs) 2: Automatically receive optional updates 3: Users can select which optional updates to receive Other value or absent: Don't receive optional updates
AllowTemporaryEnterpriseFeatureControl *Added in Windows 11, version 22H2*	REG_DWORD	1: Allowed. All features in the latest monthly cumulative update are enabled. Other value or absent: Features that are shipped turned off by default will remain off
BranchReadinessLevel	REG_DWORD	2: Systems take feature updates for the Windows Insider build - Fast 4: Systems take feature updates for the Windows Insider build - Slow 8: Systems take feature updates for the Release Windows Insider build Other value or absent: Receive all applicable updates
DeferFeatureUpdates	REG_DWORD	1: Defer feature updates Other value or absent: Don't defer feature updates
DeferFeatureUpdatesPeriodinDays	REG_DWORD	0-365: Defer feature updates by given days
DeferQualityUpdates	REG_DWORD	1: Defer quality updates Other value or absent: Don't defer quality updates

GPO Key	Key type	Value
DeferQualityUpdatesPeriodinDays	REG_DWORD	0-35: Defer quality updates by given days
ExcludeWUDriversInQualityUpdate	REG_DWORD	1: Exclude Windows Update drivers Other value or absent: Offer Windows Update drivers
PauseFeatureUpdatesStartTime	REG_DWORD	1: Pause feature updates Other value or absent: Don't pause feature updates
PauseQualityUpdatesStartTime	REG_DWORD	1: Pause quality updates Other value or absent: Don't pause quality updates

MDM: HKEY_LOCAL_MACHINE\Software\Microsoft\PolicyManager\default\Update

Expand table

MDM Key	Key type	Value
AllowOptionalContent *Added in*: • Windows 11, version 22H2 with KB5029351 and later • Windows 10, version 22H2 with KB5032278, or a later cumulative update installed	REG_DWORD	1: Automatically receive optional updates (including CFRs) 2: Automatically receive optional updates 3: Users can select which optional updates to receive Other value or absent: Don't receive optional updates
AllowTemporaryEnterpriseFeatureControl *Added in Windows 11, version 22H2*	REG_DWORD	1: Allowed. All features in the latest monthly cumulative update are enabled. Other value or absent: Features that are shipped turned off by default will remain off
BranchReadinessLevel	REG_DWORD	2: Systems take feature updates for the Windows Insider build - Fast 4: Systems take feature updates for the Windows Insider build - Slow 8: Systems take feature updates for the Release Windows Insider build 32: Systems take feature updates from General Availability Channel Note: Other value or absent: Receive all applicable updates
DeferFeatureUpdatesPeriodinDays	REG_DWORD	0-365: Defer feature updates by given days
DeferQualityUpdatesPeriodinDays	REG_DWORD	0-35: Defer quality updates by given days
ExcludeWUDriversinQualityUpdate	REG_DWORD	1: Exclude Windows Update drivers Other value or absent: Offer Windows Update drivers
PauseFeatureUpdatesStartTime	REG_DWORD	1: Pause feature updates Other value or absent: Don't pause feature updates
PauseQualityUpdatesStartTime	REG_DWORD	1: Pause quality updates Other value or absent: Don't pause quality

MDM Key	Key type	Value
		updates

Update devices to newer versions

Due to the changes in Windows Update client policies, Windows 10, version 1607 uses different GPO and MDM keys than those available in version 1511. Windows 10, version 1703 also uses a few GPO and MDM keys that are different from those available in version 1607. However, Windows Update client policies devices running older versions will still see their policies honored after they update to a newer version; the old policy keys will continue to exist with their values ported forward during the update. Following the update to a newer version, only the old keys will be populated and not the new version keys, until the newer keys are explicitly defined on the device by the administrator.

How older version policies are respected on newer versions

When a device running a newer version sees an update available on Windows Update, the device first evaluates and executes the Windows Updates for Business policy keys for its current (newer) version. If these aren't present, it then checks whether any of the older version keys are set and defer accordingly. Update keys for newer versions will always supersede the older equivalent.

Comparing keys in Windows 10, version 1607 to Windows 10, version 1703

Expand table

Version 1607 key	Version 1703 key
PauseFeatureUpdates	PauseFeatureUpdatesStartTime
PauseQualityUpdates	PauseQualityUpdatesStartTime

Last updated on 09/30/2025

Use Windows Update client policies and WSUS together

Article • 04/01/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

> **Looking for consumer information?** See Windows Update: FAQ.

The Windows update scan source policy enables you to choose what types of updates to get from either WSUS or Windows Update client policies.

We added the scan source policy starting with the September 1, 2021—KB5005101 (OS Builds 19041.1202, 19042.1202, and 19043.1202) Preview update and it applies to Window 10, version 2004 and above and Windows 11. This policy changes the way devices determine whether to scan against a local WSUS server or Windows Update service.

> **Important**
>
> - If you use Configuration Manager and have co-managed devices, see Integrate with Windows Update client policies for more information about configuring clients.
> - The policy **Do not allow update deferral policies to cause scans against Windows Update**, also known as Dual Scan, is no longer supported on Windows 11 and on Windows 10 it's replaced by the new Windows scan source policy and isn't recommended for use. If you configure both on Windows 10, you won't get updates from Windows Update.

About the scan source policy

The specify scan source policy enables you to specify whether your device gets the following Windows update types form WSUS **or** from Windows Update:

- Feature updates
- Windows quality updates
- Driver and firmware updates
- Updates for other Microsoft products

We recommend using this policy on your transition from fully on-premises managed environment to a cloud supported one. Whether you move only drivers to the cloud

today or drivers and quality updates and then later move your other workloads, taking a step-by-step approach might ease the transition.

Default scan behavior

To help you better understand the scan source policy, see the following default scan behavior and how we can change it:

- If no policies are configured: All of your updates come from Windows Update.
- If you configure only the WSUS server policy:
  - On Windows 10: All of your updates come from WSUS.
  - On Windows 11: All of your updates still come from WSUS unless you configure the specify scan source policy.
- If you configure a WSUS server and deferral policies on Windows 10: All of your updates will come from Windows Update unless you specify the scan source policy or have disabled dual scan.
- If you configure a WSUS server and the scan source policy: All of your updates will come from the source chosen in the scan source policy.

> **Tip**
>
> The only two relevant policies for where your updates come from are the specify scan source policy and whether or not you have configured a WSUS server. This should simplify the configuration options.

> **Note**
>
> If you have devices configured for WSUS and don't configure the scan source policy for feature updates to come from Windows update or set any offering policies by using Windows Update client policies, then users who select "Check online for updates" on the Settings page may see the optional upgrade to Windows 11. We recommend configuring the scan source policy or an offering policy to prevent such.

Configure the scan sources

The policy can be configured using the following two methods:

1. Group Policy: Specify source service for specific classes of Windows Updates

    - Path: Computer Configuration\Administrative Templates\Windows Components\Windows Update\Manage updates offered from Windows Server Update Service\

2. Configuration Service Provider (CSP) Policies:
   SetPolicyDrivenUpdateSourceFor<Update Type>:

> ⓘ **Note**
>
> - You should configure **all** of these policies if you're using CSPs.
> - Editing the registry to change the behavior of update policies isn't recommended. Use Group Policy or the Configuration Service Provider (CSP) policy instead of directly writing to the registry. However, if you choose to edit the registry, ensure you've configured the `UseUpdateClassPolicySource` registry key too, or the scan source won't be altered.

- If you're also using the **Specify settings for optional component installation and component repair** policy to enable content for FoDs and language packs, see How to make Features on Demand and language packs available when you're using WSUS or Configuration Manager to verify your policy configuration.

- Update/SetPolicyDrivenUpdateSourceForDriverUpdates
- Update/SetPolicyDrivenUpdateSourceForFeatureUpdates
- Update/SetPolicyDrivenUpdateSourceForOtherUpdates
- Update/SetPolicyDrivenUpdateSourceForQualityUpdates

Feedback

Was this page helpful? Yes No

Provide product feedback

Enforcing compliance deadlines for updates

Article • 03/18/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

Deploying feature or quality updates for many organizations is only part of the equation for managing their device ecosystem. The ability to enforce update compliance is the next important part. Windows Update client policies provide controls to manage deadlines for when devices should migrate to newer versions. This article contains information on how to enforce compliance deadlines for clients that use Windows Update client policies.

Policies for compliance deadlines

Policies for Windows 11, version 22H2 and later

Policies for clients running Windows 11, version 22H2 and later

With Windows 11, version 22H2 and later, the following policies are available to manage compliance deadlines for updates:

Expand table

Policy	Description
Specify deadline for automatic updates and restarts for quality update	This policy lets you specify the number of days before quality updates are installed on devices automatically, and a grace period, after which required restarts occur automatically. This policy includes an option to opt out of automatic restarts until the end of the grace period is reached.
Specify deadline for automatic updates and restarts for feature update	This policy lets you specify the number of days before feature updates are installed on devices automatically, and a grace period, after which required restarts occur automatically. This policy includes an option to opt out of automatic restarts until the end of the grace period is reached.

In MDM, these policies are available as separate settings:

- Update/ConfigureDeadlineForFeatureUpdates

- Update/ConfigureDeadlineForQualityUpdates
- Update/ConfigureDeadlineGracePeriod (for quality updates)
- Update/ConfigureDeadlineGracePeriodForFeatureUpdates
- Update/ConfigureDeadlineNoAutoRebootForQualityUpdates
- Update/ConfigureDeadlineNoAutoRebootForFeatureUpdates

When **Specify deadline for automatic updates and restarts** for either quality updates or feature updates is set:

The deadline calculation for both quality and feature updates is based off the time the client's update scan initially discovered the update. Previously, the deadline was based off the release date of the update for quality updates and the reboot pending date for feature updates. The change for deadline calculation was made to improve the predictability of restart.

The grace period for both quality and feature updates starts its countdown from the time of a pending restart after the installation is complete. This grace period is especially helpful for users returning from vacation or the time away, preventing an immediate forced reboot when they come back.

The *effective deadline* is whichever is the later of the scan discovery time plus the specified deadline or the scan discovery time plus the grace period. As soon as installation is complete and the device reaches pending restart, users are able to schedule restarts before effective deadline and Windows can still automatically restart outside of active hours if users choose not to schedule restarts. Once the effective deadline is reached, the device tries to restart regardless of active hours.

> **Note**
>
> - When these policies are used, user settings for notifications are also used on clients running Windows 11, version 22H2 and later.
> - When **Specify deadline for automatic updates and restarts** for either quality updates or feature updates is used, updates will be downloaded and installed as soon as they are offered.
> - When **Specify deadline for automatic updates and restarts** for either quality updates or feature updates is used, download, installation, and reboot settings stemming from the Configure Automatic Updates are ignored.
> - Starting with the December 10, 2024 update for Windows 11, version 22H2 and later clients, Configure Automatic Updates are respected before the deadline occurs, and ignored once the deadline passes. For

instance, if you set up Configure Automatic Updates to schedule update installation at 3:00 AM, you also set up a commercial deadline, then the download and install occurs at the scheduled time from Configure Automatic Updates so long as it's not past the deadline.

User experience for restart notifications with compliance deadlines

These deadline policies also offer an option to opt out of automatic restarts until a deadline is reached by presenting an "engaged restart experience" until the deadline passes. At that point, the device automatically schedules a restart regardless of active hours.

These notifications are what the user sees depending on the settings you choose, and what operating system version their device is running. Generally, the user notifications become more noticeable as the deadline approaches. The experience described is the default and assumes there's ample time for notifications before the effective deadline occurs. The description doesn't account for changes to the **Display options for update notifications** policy (Update/NoUpdateNotificationsDuringActiveHours) or other settings that would significantly change the experience.

Windows 11, version 23H2 and later

The following notifications are what the user sees on Windows 11, version 23H2 and later, depending on the settings chosen by the user and the IT administrator:

When **Specify deadlines for automatic updates and restarts** is set:

While restart is pending, before the deadline occurs, users receive a toast notification in the corner of their screen. The notification includes the deadline date, and options to either restart now, pick a time to restart, or restart tonight once active hours ends.

- If the user set the option **Settings** > **Windows Update** > **Advanced options** > **Notify me when a restart is required to finish updating** to **On**, they immediately receive the toast notification when the device enters a restart pending state for updates. Automatic restarts for updates are blocked for 24 hours after the initial notification to give these users time to prepare.

- If the user set **Notify me when a restart is required to finish updating** to **Off** (default), they receive a toast notification that a restart is required 24 hours after the device enters a restart pending state for updates.

Depending on settings both users and admins configure, toast notification may occur occasionally before the day of the deadline to remind the user of the update. During this time, if they're allowed, automatic restarts might be scheduled after active hours.

- If an automatic restart is scheduled or the user scheduled the restart, and the user is signed in at that time, they receive a notification 15 minutes before the scheduled time.

As the device approaches the deadline time, a notification displays in the middle of the screen that contains the deadline time and options to restart now or acknowledge the notification.

15 minutes before the deadline, a notification displays in the middle of the screen notifying the user that a restart is going to occur. Users can either confirm the restart, reschedule, or choose to restart now.

In cases where a user scheduled restart fails but there's still more time before the deadline is reached, the user receives a notification to either restart now or to reschedule the restart.

In cases where the deadline has passed, the user receives a notification that a restart is required. The only options a user can select is to restart now or confirm. The user has 15 minutes to select restart before the device is forced to restart.

In cases where the deadline has passed and the restart failed, the user receives a notification that a restart is required. If the device is plugged in, it will attempt to restart every 5 minutes until the device successfully restarts. The user has 5 minutes to restart before the device is forced to restart.

Feedback

Was this page helpful? Yes No

Provide product feedback

Integrate Windows Update client policies with management solutions

Article • 03/31/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

> **Looking for consumer information?** See Windows Update: FAQ.

You can integrate Windows Update client policy deployments with existing management tools such as Windows Server Update Services (WSUS) and Microsoft Configuration Manager.

> **Tip**
>
> This feature was formerly known as *Windows Update for Business*.

Integrate Windows Update client policies with Windows Server Update Services

For Windows 10, version 1607 and later, devices can be configured to receive updates from both Windows Update (or Microsoft Update) and Windows Server Update Services (WSUS). When using both WSUS and Windows Update client policies:

- Devices receive their Windows content from Microsoft and defer these updates according to Windows Update client policies
- All other content synced from WSUS will be directly applied to the device; that is, updates to products other than Windows won't follow your deferral policies.

Configuration example #1: Deferring Windows Update updates with other update content hosted on WSUS

Configuration:

- Device is configured to defer Windows quality updates using Windows Update client policies
- Device is also configured to be managed by WSUS
- Device isn't configured to enable Microsoft Update (**Update/AllowMUUpdateService** = not enabled)
- Admin has opted to put updates to Office and other products on WSUS
- Admin has also put third-party drivers on WSUS

Expand table

Content	Metadata source	Payload source	Deferred?
Updates to Windows	Windows Update	Windows Update	Yes
Updates to Office and other products	WSUS	WSUS	No
Third-party drivers	WSUS	WSUS	No

Configuration example #2: Excluding drivers from Windows quality updates using Windows Update client policies

Configuration:

- Device is configured to defer Windows quality updates and to exclude drivers from Windows Update quality updates (**ExcludeWUDriversInQualityUpdate** = enabled)
- Device is also configured to be managed by WSUS
- Admin has opted to put Windows Update drivers on WSUS

Expand table

Content	Metadata source	Payload source	Deferred?
Updates to Windows (excluding drivers)	Windows Update	Windows Update	Yes
Updates to Office and other products	WSUS	WSUS	No
Drivers	WSUS	WSUS	No

Configuration example #3: Device configured to receive Microsoft updates

Configuration:

- Device is configured to defer quality updates using Windows Update client policies and to be managed by WSUS
- Device is configured to **receive updates for other Microsoft products** along with updates to Windows (**Update/AllowMUUpdateService** = enabled)
  - For a list of other Microsoft products that might be updated, see Update other Microsoft products.
- Admin has also placed Microsoft Update, non-Microsoft, and locally published update content on the WSUS server

In this example, the deferral behavior for updates to Office and other non-Windows products is slightly different than if WSUS weren't enabled.

- In a non-WSUS case, these updates would be deferred just as any update to Windows would be.
- However, with WSUS also configured, these updates are sourced from Microsoft but deferral policies aren't applied.

Expand table

Content	Metadata source	Payload source	Deferred?
Updates to Windows (excluding drivers)	Microsoft Update	Microsoft Update	Yes
Updates to Office and other products	Microsoft Update	Microsoft Update	No
Drivers, third-party applications	WSUS	WSUS	No

> **Note**
>
> Because the admin enabled **Update/AllowMUUpdateService**, placing the content on WSUS was not needed for the particular device, as the device will always receive Microsoft Update content from Microsoft when configured in this manner.

Integrate Windows Update client policies with Microsoft Configuration Manager

For Windows 10, version 1607, organizations already managing their systems with a Configuration Manager solution can also have their devices configured for Windows Update client policies (that is, setting deferral policies on those devices). Such devices are visible in the Configuration Manager console, however they appear with a detection state of **Unknown**.

For more information, see Integration with Windows Update client policies.

Feedback

Was this page helpful? Yes No

Provide product feedback

Walkthrough: Use Group Policy to configure Windows Update client policies

Applies to: ✅ Windows 11, ✅ Windows 10, ✅ Windows Server 2022, ✅ Windows Server 2019, ✅ Windows Server 2016

> **Looking for consumer information?** See Windows Update: FAQ.

Overview

You can use Group Policy through the Group Policy Management Console (GPMC) to control how Windows Update client policies work. You should consider and devise a deployment strategy for updates before you make changes to the Windows Update client policies. For more information, see Prepare servicing strategy for Windows client updates for more information.

An IT administrator can configure Windows Update client policies by using Group Policy, or they can be set locally (per device). All of the relevant policies are under the path **Computer configuration** > **Administrative Templates** > **Windows Components** > **Windows Update**.

> **Tip**
>
> This feature was formerly known as *Windows Update for Business*.

To manage updates with Windows Update client policies as described in this article, you should prepare with these steps, if you haven't already:

- Create Active Directory security groups that align with the deployment rings you use to phase deployment of updates.
- Allow access to the Windows Update service.
- Download and install ADMX templates appropriate to your Windows 10 version. For more information, see How to create and manage the Central Store for Group Policy Administrative Templates in Windows and Step-By-Step: Managing Windows 10 with Administrative templates.

Set up Windows Update client policies

In this example, one security group is used to manage updates. Typically we would recommend having at least three rings (early testers for pre-release builds, broad deployment for releases, critical devices for mature releases) to deploy.

Follow these steps on a device running the Remote Server Administration Tools or on a domain controller:

Set up a ring

1. Start Group Policy Management Console (gpmc.msc).
2. Expand **Forest > Domains > <*your domain*>.
3. Right-click <*your domain*> and select **Create a GPO in this domain and link it here**.
4. In the **New GPO** dialog box, enter **Windows Update client policies - Group 1** as the name of the new Group Policy Object.
5. Right-click the **"Windows Update client policies - Group 1"** object, and then select **Edit**.
6. In the Group Policy Management Editor, go to **Computer Configuration** > **Policies** > **Administrative Templates** > **Windows Components** > **Windows Update**. You're now ready to start assigning policies to this ring (group) of devices.

Manage Windows Update offerings

You can control when updates are applied, for example by deferring when an update is installed on a device or by pausing updates for a certain period of time.

Determine which updates you want offered to your devices

Both feature and quality updates are automatically offered to devices that are connected to Windows Update using Windows Update client policies. However, you can choose whether you want the devices to additionally receive other Microsoft Updates or drivers that are applicable to that device.

To enable Microsoft Updates, use the Group Policy Management Console go to **Computer Configuration** > **Administrative Templates** > **Windows Components** > **Windows Update** > **Configure Automatic Updates** and select **Install updates for other Microsoft products**. For a list of other Microsoft products that might be updated, see Update other Microsoft products.

Drivers are automatically enabled because they're beneficial to device systems. We recommend that you allow the driver policy to allow drivers to update on devices (the default), but you can turn off this setting if you prefer to manage drivers manually. If you want to disable driver updates for some reason, use the Group Policy Management Console to go to **Computer Configuration** > **Administrative Templates** > **Windows Components** > **Windows Update** > **Do not include drivers with Windows Updates** and enable the policy.

We also recommend that you allow Microsoft product updates as discussed previously.

Set when devices receive feature and quality updates

I want to receive pre-release versions of the next feature update

1. Ensure that you're enrolled in the Windows Insider Program for Business. This is a free program available to commercial customers to aid them in their validation of feature updates before they're released. Joining the program enables you to receive updates prior to their release and receive emails and content related to what is coming in the next updates.

2. Use Group Policy Management Console to go to: **Computer Configuration** > **Administrative Templates** > **Windows Components** > **Windows Update** > **Windows Update for Business** > **Manage preview builds** and set the policy to **Enable preview builds** for any of test devices you want to install pre-release builds.

3. Use Group Policy Management Console to go to **Computer Configuration** > **Administrative Templates** > **Windows Components** > **Windows Update** > **Windows Update for Business** > **Select when Preview Builds and feature updates are received**. In the **Options** pane, use the pulldown menu to select one of the preview builds. We recomment **Windows Insider Program Slow** for commercial customers using pre-release builds for validation.

4. Select **OK**.

I want to manage which released feature update my devices receive

A Windows Update administrator can defer or pause updates. You can defer feature updates for up to 365 days and defer quality updates for up to 30 days. Deferring simply means that you won't receive the update until it has been released for at least the number of deferral days you specified (offer date = release date + deferral date). You can pause feature or quality updates for up to 35 days from a given start date that you specify.

- To defer or pause a feature update: **Computer configuration** > **Administrative Templates** > **Windows Components** > **Windows Update** > **Windows Update for Business** > **Select when Preview Builds and feature updates are Received**
- Defer or pause a quality update: **Computer configuration** > **Administrative Templates** > **Windows Components** > **Windows Update** > **Windows Update for Business** > **Select when Quality Updates are Received**

Example

In this example, there are three rings for quality updates. The first ring ("pilot") has a deferral period of 0 days. The second ring ("fast") has a deferral of five days. The third ring ("slow") has a deferral of 10 days.

When the quality update is released, it's offered to devices in the pilot ring the next time they scan for updates.

Five days later

The devices in the fast ring are offered the quality update the next time they scan for updates.

Ten days later

Ten days after the quality update is released, it's offered to the devices in the slow ring the next time they scan for updates.

If no problems occur, all of the devices that scan for updates will be offered the quality update within 10 days of its release, in three waves.

What if a problem occurs with the update?

In this example, some problem is discovered during the deployment of the update to the "pilot" ring.

At this point, the IT administrator can set a policy to pause the update. In this example, the admin selects the **Pause quality updates** check box.

Now all devices are paused from updating for 35 days. When the pause is removed, they'll be offered the *next* quality update, which ideally won't have the same issue. If there's still an issue, the IT admin can pause updates again.

I want to stay on a specific version

If you need a device to stay on a version beyond the point when deferrals on the next version would elapse or if you need to skip a version, use the **Select the target feature update version** setting instead of using the **Specify when Preview Builds and feature updates are received** setting for feature update deferrals. When you use this policy, specify the version that you want your devices to use. If you don't update this before the device reaches end of service, the device will automatically be updated once it's 60 days past end of service for its edition.

When you set the target version policy, if you specify a feature update version that is older than your current version or set a value that isn't valid, the device won't receive any feature updates until the policy is updated. When you specify target version policy, feature update deferrals won't be in effect.

Manage how users experience updates

I want to manage when devices download, install, and restart after updates

We recommend that you allow to update automatically--this is the default behavior. If you don't set an automatic update policy, the device will attempt to download, install, and restart at the best times for the user by using built-in intelligence such as intelligent active hours.

For more granular control, you can set the maximum period of active hours the user can set with **Computer Configuration** > **Administrative Templates** > **Windows Components** > **Windows Update** > **Specify active hours range for auto restart**.

It's best to refrain from setting the active hours policy because it's enabled by default when automatic updates aren't disabled and provides a better experience when users can set their own active hours. If you do want to set active hours, use **Computer Configuration** > **Administrative Templates** > **Windows Components** > **Windows Update** > **Turn off auto-restart for updates during active hours**.

To update outside of the active hours, you don't need to set any additional settings: simply don't disable automatic restarts. For even more granular control, consider using automatic updates to schedule the install time, day, or week. To do this, use **Computer Configuration** > **Administrative Templates** > **Windows Components** > **Windows Update** > **Configure Automatic Updates** and select **Auto download and schedule the install**. You can customize this setting to accommodate the time that you want the update to be installed for your devices.

When you set these policies, installation happens automatically at the specified time and the device will restart 15 minutes after installation is complete (unless it's interrupted by the user).

I want to keep devices secure and compliant with update deadlines

We recommend that you use **Computer Configuration** > **Administrative Templates** > **Windows Components** > **Windows Update** > **Specify deadline for automatic updates and restarts** for feature and quality updates to ensure that devices stay secure on Windows 10, version 1709 and later. This works by enabling you to specify the number of days that can elapse after an update is offered to a device before it must be installed. Also you can set the number of days that can elapse after a pending restart before the user is forced to restart.

For more information, see Enforcing compliance deadlines for updates.

End user settings for notifications

Applies to:

- Windows 11, version 23H2 with KB5037771 or later
- Windows 11, version 22H2 with KB5037771 or later

Users can set a preference for notifications about pending restarts for updates under **Settings** > **Windows Update** > **Advanced options** > **Notify me when a restart is required to finish updating**. This setting is end-user controlled and not controlled or configurable by IT administrators.

Users have the following options for the **Notify me when a restart is required to finish updating** setting:

- **Off** (default): Once the device enters a pending reboot state for updates, restart notifications are suppressed for 24 hours. During the first 24 hours, automatic restarts can still occur outside of active hours. Typically, users receive fewer notifications about upcoming restarts while the deadline is approaching.
  - When the deadline is set for 1 day, users only receive a notification about the deadline and a final nondismissable notification 15 minutes before a forced restart.

- **On**: Users immediately receive a toast notification when the device enters a reboot pending state for updates. Automatic restarts for updates are blocked for 24 hours after the initial notification to give these users time to prepare for a restart. After 24 hours have passed, automatic restarts can occur. This setting is recommended for users who want to be notified about upcoming restarts.

- When the deadline is set for 1 day, an initial notification occurs, automatic restart is blocked for 24 hours, and users receive another notification before the deadline and a final nondismissable notification 15 minutes before a forced restart.

When a deadline is set for 0 days, no matter which option is selected, the only notification users receive is a final nondismissable notification 15 minutes before a forced restart.

The user preference for notifications applies when compliance deadlines are used. The policy for compliance deadlines is under **Computer Configuration** > **Administrative Templates** > **Windows Components** > **Windows Update** > **Specify deadlines for automatic updates and restarts**.

- In Windows version 22H2 and later, the **Specify deadlines for automatic updates and restarts** policy is split in to two policies: **Specify deadline for automatic updates and restarts for quality update** and **Specify deadline for automatic updates and restarts for feature update**.

I want to manage the notifications a user sees

There are additional settings that affect the notifications.

We recommend that you use the default notifications as they aim to provide the best user experience while adjusting for the compliance policies that you have set. If you do have further needs that aren't met by the default notification settings, you can use **Computer Configuration** > **Administrative Templates** > **Windows Components** > **Windows Update** > **Display options for update notifications** with these values:

0 (default) - Use the default Windows Update notifications
1 - Turn off all notifications, excluding restart warnings
2 - Turn off all notifications, including restart warnings

Option **2** creates a poor experience for personal devices; it's only recommended for kiosk devices where automatic restarts have been disabled.

> **Note**
>
> Starting in Windows 11, version 22H2, **Apply only during active hours** was added as an additional option for **Display options for update notifications**. When **Apply only during active hours** is selected, the notifications will only be disabled during active hours when options `1` or `2` are used. To ensure that the device stays updated, a notification will still be shown during active hours if **Apply only during active hours** is selected, and once a

deadline has been reached when Specify deadlines for automatic updates and restarts is configured.

Still more options are available in **Computer Configuration** > **Administrative Templates** > **Windows Components** > **Windows Update** > **Configure auto-restart restart warning notifications schedule for updates**. This setting allows you to specify the period for auto-restart warning reminder notifications (from 2-24 hours; 4 hours is the default) before the update and to specify the period for auto-restart imminent warning notifications (15-60 minutes is the default). We recommend using the default notifications. This policy is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

I want to manage the update settings a user can access

Every Windows device provides users with various controls they can use to manage Windows Updates. They can access these controls by Search to find Windows Updates or by going selecting **Updates and Security** in **Settings**. We provide the ability to disable a variety of these controls that are accessible to users.

Users with access to update pause settings can prevent both feature and quality updates for 7 days. You can prevent users from pausing updates through the Windows Update settings page by using **Computer Configuration** > **Administrative Templates** > **Windows Components** > **Windows Update** > **Remove access to Pause updates**. When you disable this setting, users will see **Some settings are managed by your organization** and the update pause settings are greyed out.

If you use Windows Server Update Server (WSUS), you can prevent users from scanning Windows Update. To do this, use **Computer Configuration** > **Administrative Templates** > **Windows Components** > **Windows Update** > **Remove access to use all Windows Update features**.

I want to enable optional updates

Applies to:

- Windows 11, version 22H2 with KB5029351 and later
- Windows 10, version 22H2 with KB5032278, or a later cumulative update installed

In addition to the monthly cumulative update, optional updates are available to provide new features and nonsecurity changes. Most optional updates are released on the fourth Tuesday of the month, known as optional nonsecurity preview releases. Optional updates can also include features that are gradually rolled out, known as controlled feature rollouts (CFRs).

Installation of optional updates isn't enabled by default for devices that receive updates using Windows Update client policies. However, you can enable optional updates for devices by using the **Computer Configuration** > **Administrative Templates** > **Windows Components** > **Windows Update** > **Manage updates offered from Windows Update** > **Enable optional updates** policy.

To keep the timing of updates consistent, the **Enable optional updates** policy respects the deferral period for quality updates. This policy allows you to choose if devices should receive CFRs in addition to the optional nonsecurity preview releases, or if the end-user can make the decision to install optional updates. This policy can change the behavior of the **Get the latest updates as soon as they're available** option in **Settings** > **Update & security** > ***Windows Update** > **Advanced options**.

The following options are available for the policy:

- **Automatically receive optional updates (including CFRs)**:
  - The latest optional nonsecurity updates and CFRs are automatically installed on the device. The quality update deferral period is applied to the installation of these updates.
  - The **Get the latest updates as soon as they're available** option is selected and users can't change the setting.
  - Devices will receive CFRs in early phases of the rollout.

- **Automatically receive optional updates**:
  - The latest optional nonsecurity updates are automatically installed on the device but CFRs aren't. The quality update deferral period is applied to the installation of these updates.
  - The **Get the latest updates as soon as they're available** option isn't selected and users can't change the setting.

- **Users can select which optional updates to receive**:
  - Users can select which optional updates to install from **Settings** > **Update & security** > **Windows Update** > **Advanced options** > **Optional updates**.
    - Optional updates are offered to the device, but user interaction is required to install them unless the **Get the latest updates as soon as they're available** option is also enabled.
    - CFRs are offered to the device, but not necessarily in the early phases of the rollout.
  - Users can enable the **Get the latest updates as soon as they're available** option in **Settings** > **Update & security** > ***Windows Update** > **Advanced options**. If the user enables the **Get the latest updates as soon as they're available**, then:
    - The device will receive CFRs in early phases of the rollout.
    - Optional updates are automatically installed on the device.

- **Not configured** (default):
  - Optional updates aren't installed on the device and the **Get the latest updates as soon as they're available** option is disabled.

I want to enable features introduced via servicing that are off by default

(*Starting in Windows 11, version 22H2 or later*)

New features and enhancements are introduced through the monthly cumulative update to provide continuous innovation for Windows 11. To give organizations time to plan and prepare, some of these new features are temporarily turned off by default. Features that are turned off by default are listed in the KB article for the monthly cumulative update. Typically, a feature is selected to be off by default because it either impacts the user experience or IT administrators significantly.

The features that are turned off by default from servicing updates will be enabled in the next annual feature update. Organizations can choose to deploy feature updates at their own pace, to delay these features until they're ready for them.

You can enable these features by using **Computer Configuration** > **Administrative Templates** > **Windows Components** > **Windows Update** > **Manage end user experience** > **Enable features introduced via servicing that are off by default**. The following options are available:

- **Enabled**: All features in the latest monthly cumulative update are enabled.
  - When the policy is set to **Enabled**, all features that are currently turned off will turn on when the device next reboots
- **Disabled** - Features that are shipped turned off by default will remain off
- **Not configured** - Features that are shipped turned off by default will remain off

---

Last updated on 09/26/2025

Walkthrough: Use CSPs and MDMs to configure Windows Update client policies

Article • 03/18/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

> **Looking for consumer information?** See Windows Update: FAQ.

Overview

You can use Configuration Service Provider (CSP) policies to control how Windows Update client policies work by using a Mobile Device Management (MDM) tool. You should consider and devise a deployment strategy for updates before you make changes to the Windows Update client policies.

> **Tip**
>
> This feature was formerly known as *Windows Update for Business*.

An IT administrator can configure Windows Update client policies by using Microsoft Intune or a non-Microsoft MDM tool.

To manage updates with Windows Update client policies, you should prepare with these steps, if you haven't already:

- Create Active Directory security groups that align with the deployment rings you use to phase deployment of updates. See Build deployment rings for Windows client updates to learn more about deployment rings in Windows client.
- Allow access to the Windows Update service.

Manage Windows Update offerings

You can control when updates are applied, for example by deferring when an update is installed on a device or by pausing updates for a certain period of time.

Determine which updates you want offered to your devices

Both feature and quality updates are automatically offered to devices that are connected to Windows Update using Windows Update client policies. However, you can choose whether you want the devices to additionally receive other Microsoft Updates or drivers that are applicable to that device.

To enable Microsoft Updates, use Update/AllowMUUpdateService.

Drivers are automatically enabled because they're beneficial to device systems. We recommend that you allow the driver policy to allow drivers to be updated on devices (the default), but you can turn off this setting if you prefer to manage drivers manually. If you want to disable driver updates for some reason, use Update/ExcludeWUDriversInQualityUpdate.

We also recommend that you allow Microsoft product updates as discussed previously.

Set when devices receive feature and quality updates

I want to receive prerelease versions of the next feature update

1. Ensure that you're enrolled in the Windows Insider Program for Business. Windows Insider is a free program available to commercial customers to aid them in their validation of feature updates before they're released. Joining the program enables you to receive updates prior to their release as well as receive emails and content related to what is coming in the next updates.

2. For any of test devices you want to install prerelease builds, use Update/ManagePreviewBuilds. Set the option to **Enable preview builds**.

3. Use Update/BranchReadinessLevel and select one of the preview Builds. Windows Insider Program Slow is the recommended channel for commercial customers who are using prerelease builds for validation.

4. Additionally, you can defer prerelease feature updates the same way as released updates, by setting a deferral period up to 14 days by using Update/DeferFeatureUpdatesPeriodInDays. If you're testing with Windows Insider Program Slow builds, we recommend that you receive the preview updates to your IT department on day 0, when the update is released, and then have a 7-10 day deferral before rolling out to your group of testers. This schedule helps ensure that if a problem is discovered, you can pause the rollout of the preview update before it reaches your tests.

I want to manage which released feature update my devices receive

A Windows Update administrator can defer or pause updates. You can defer feature updates for up to 365 days and defer quality updates for up to 30 days. Deferring simply means that you don't receive the update until it has been released for at least the number of deferral days you specified (offer date = release date + deferral date). You can pause feature or quality updates for up to 35 days from a given start date that you specify.

- To defer a feature update: Update/DeferFeatureUpdatesPeriodInDays
- To pause a feature update: Update/PauseFeatureUpdatesStartTime
- To defer a quality update: Update/DeferQualityUpdatesPeriodInDays
- To pause a quality update: Update/PauseQualityUpdatesStartTime

Example

In this example, there are three rings for quality updates. The first ring ("pilot") has a deferral period of 0 days. The second ring ("fast") has a deferral of five days. The third ring ("slow") has a deferral of ten days.

When the quality update is released, it's offered to devices in the pilot ring the next time they scan for updates.

Five days later

The devices in the fast ring are offered the quality update the next time they scan for updates.

Ten days later

Ten days after the quality update is released, it's offered to the devices in the slow ring the next time they scan for updates.

If no problems occur, all of the devices that scan for updates are offered the quality update within ten days of its release, in three waves.

What if a problem occurs with the update?

In this example, some problem is discovered during the deployment of the update to the "pilot" ring.

At this point, the IT administrator can set a policy to pause the update. In this example, the admin selects the **Pause quality updates** check box.

Now all devices are paused from updating for 35 days. When the pause is removed, they'll be offered the *next* quality update, which ideally won't have the same issue. If there's still an issue, the IT admin can pause updates again.

I want to stay on a specific version

If you need a device to stay on a version beyond the point when deferrals on the next version would elapse or if you need to skip a version (for example, update fall release to fall release) use the Update/TargetReleaseVersion (or Deploy Feature Updates Preview in Intune) instead of using feature update deferrals. When you use this policy, specify the version that you want your device(s) to move to or stay on (for example, "1909"). You can find version information at the Windows 10 Release Information Page.

Manage how users experience updates

I want to manage when devices download, install, and restart after updates

We recommended that you allow updating automatically, which is the default behavior. If you don't set an automatic update policy, the device attempts to download, install, and restart at the best times for the user by using built-in intelligence such as intelligent active hours.

For more granular control, you can set the maximum period of active hours the user can set with Update/ActiveHoursMaxRange. You could also set specific start and end times for active ours with Update/ActiveHoursEnd and Update/ActiveHoursStart.

It's best to refrain from setting the active hours policy because it's enabled by default when automatic updates aren't disabled and provides a better experience when users can set their own active hours.

To update outside of the active hours, use Update/AllowAutoUpdate with Option 2 (which is the default setting). For even more granular control, consider using automatic updates to schedule the install time, day, or week. To use a schedule, use Option 3, and then set the following policies as appropriate for your plan:

- Update/ScheduledInstallDay
- Update/ScheduledInstallEveryWeek
- Update/ScheduledInstallFirstWeek
- Update/ScheduledInstallFourthWeek
- Update/ScheduledInstallSecondWeek
- Update/ScheduledInstallThirdWeek
- Update/ScheduledInstallTime

When you set these policies, installation happens automatically at the specified time and the device will restart 15 minutes after installation is complete (unless it's interrupted by the user).

If you don't want to allow any automatic updates prior to the deadline, set Update/AllowAutoUpdate to Option 5, which turns off automatic updates.

I want to keep devices secure and compliant with update deadlines

We recommend that you use set specific deadlines for feature and quality updates to ensure that devices stay secure on Windows 10, version 1709 and later. Deadlines work by enabling you to specify the number of days that can elapse after an update is offered to a device before it must be installed. Also you can set the number of days that can elapse after a pending restart before the user is forced to restart. For more information about these settings, see Enforcing compliance deadlines for updates. The following settings enforce compliance deadlines for updates:

For Windows 10, version 22H2:

- Update/ConfigureDeadlineForFeatureUpdates
- Update/ConfigureDeadlineForQualityUpdates
- Update/ConfigureDeadlineGracePeriod
- Update/ConfigureDeadlineGracePeriodForFeatureUpdates
- Update/ConfigureDeadlineNoAutoReboot

For Windows 11, version 22H2 and later:

- Update/ConfigureDeadlineForFeatureUpdates
- Update/ConfigureDeadlineForQualityUpdates
- Update/ConfigureDeadlineGracePeriod (for quality updates)
- Update/ConfigureDeadlineGracePeriodForFeatureUpdates
- Update/ConfigureDeadlineNoAutoRebootForQualityUpdates
- Update/ConfigureDeadlineNoAutoRebootForFeatureUpdates

> **Note**
>
> - When these policies are used, user settings for notifications are also used on clients running Windows 11, version 22H2 and later.
> - When **Specify deadlines for automatic updates and restarts** for either quality updates or feature updates is used, updates will be downloaded and installed as soon as they are offered.
> - When **Specify deadlines for automatic updates and restarts** for either quality updates or feature updates is used, download, installation, and reboot settings stemming from the Configure Automatic Updates are ignored.

- Starting with the December 10, 2024 update for Windows 11, version 22H2 and later clients, Configure Automatic Updates are respected before the deadline occurs, and ignored once the deadline passes.

End user settings for notifications

Applies to:

- Windows 11, version 23H2 with KB5037771 or later
- Windows 11, version 22H2 with KB5037771 or later

Users can set a preference for notifications about pending restarts for updates under **Settings** > **Windows Update** > **Advanced options** > **Notify me when a restart is required to finish updating**. This setting is end-user controlled and not controlled or configurable by IT administrators.

Users have the following options for the **Notify me when a restart is required to finish updating** setting:

- **Off** (default): Once the device enters a pending reboot state for updates, restart notifications are suppressed for 24 hours. During the first 24 hours, automatic restarts can still occur outside of active hours. Typically, users receive fewer notifications about upcoming restarts while the deadline is approaching.
  - When the deadline is set for 1 day, users only receive a notification about the deadline and a final nondismissable notification 15 minutes before a forced restart.

- **On**: Users immediately receive a toast notification when the device enters a reboot pending state for updates. Automatic restarts for updates are blocked for 24 hours after the initial notification to give these users time to prepare for a restart. After 24 hours have passed, automatic restarts can occur. This setting is recommended for users who want to be notified about upcoming restarts.
  - When the deadline is set for 1 day, an initial notification occurs, automatic restart is blocked for 24 hours, and users receive another notification before the deadline and a final nondismissable notification 15 minutes before a forced restart.

When a deadline is set for 0 days, no matter which option is selected, the only notification users receive is a final nondismissable notification 15 minutes before a forced restart.

The user preference for notifications applies when the following policies for compliance deadlines are used:

For Windows 10, version 22H2:

- Update/ConfigureDeadlineForFeatureUpdates
- Update/ConfigureDeadlineForQualityUpdates
- Update/ConfigureDeadlineGracePeriod
- Update/ConfigureDeadlineGracePeriodForFeatureUpdates
- Update/ConfigureDeadlineNoAutoReboot

For Windows 11, version 22H2 and later:

- Update/ConfigureDeadlineForFeatureUpdates
- Update/ConfigureDeadlineForQualityUpdates
- Update/ConfigureDeadlineGracePeriod (for quality updates)
- Update/ConfigureDeadlineGracePeriodForFeatureUpdates
- Update/ConfigureDeadlineNoAutoRebootForQualityUpdates
- Update/ConfigureDeadlineNoAutoRebootForFeatureUpdates

I want to manage the notifications a user sees

There are additional settings that affect the notifications.

We recommend that you use the default notifications as they aim to provide the best user experience while adjusting for the compliance policies that you set. If you do have further needs that aren't met by the default notification settings, you can use the Update/NoUpdateNotificationsDuringActiveHours policy with these values:

0 (default) - Use the default Windows Update notifications
1 - Turn off all notifications, excluding restart warnings
2 - Turn off all notifications, including restart warnings

> ⓘ **Note**
>
> Option **2** creates a poor experience for personal devices; it's only recommended for kiosk devices where automatic restarts have been disabled.

Still more options are available in Update/ScheduleRestartWarning. This setting allows you to specify the period for auto restart warning reminder notifications (from 2-24 hours; 4 hours is the default) before the update. You can also specify the period for auto

restart imminent warning notifications with Update/ScheduleImminentRestartWarning (15-60 minutes is the default). We recommend using the default notifications.

I want to manage the update settings a user can access

Every Windows device provides users with various controls they can use to manage Windows Updates. They can access these controls by Search to find Windows Updates or by going selecting **Updates and Security** in **Settings**. We provide the ability to disable a variety of these controls that are accessible to users.

Users with access to update pause settings can prevent both feature and quality updates for 7 days. You can prevent users from pausing updates through the Windows Update settings page by using Update/SetDisablePauseUXAccess. When you disable this setting, users see **Some settings are managed by your organization** and the update pause settings are greyed out.

If you use Windows Server Update Server (WSUS), you can prevent users from scanning Windows Update. To do this, use Update/SetDisableUXWUAccess.

I want to enable features introduced via servicing that are off by default

(*Starting in Windows 11, version 22H2 or later*)

New features and enhancements are introduced through the monthly cumulative update to provide continuous innovation for Windows 11. To give organizations time to plan and prepare, some of these new features are temporarily turned off by default. Features that are turned off by default are listed in the KB article for the monthly cumulative update. Typically, a feature is selected to be off by default because it either impacts the user experience or IT administrators significantly.

The features that are turned off by default from servicing updates will be enabled in the next annual feature update. Organizations can choose to deploy feature updates at their own pace, to delay these features until they're ready for them.

You can enable these features by using AllowTemporaryEnterpriseFeatureControl. The following options are available:

- **0** (default): Not allowed. Features that are shipped turned off by default will remain off
- **1**: Allowed. All features in the latest monthly cumulative update are enabled.

- When the policy is set to **1**, all features that are currently turned off will turn on when the device next reboots.

I want to enable optional updates

Applies to:

- Windows 11, version 22H2 with KB5029351 and later
- Windows 10, version 22H2 with KB5032278, or a later cumulative update installed

In addition to the monthly cumulative update, optional updates are available to provide new features and nonsecurity changes. Most optional updates are released on the fourth Tuesday of the month, known as optional nonsecurity preview releases. Optional updates can also include features that are gradually rolled out, known as controlled feature rollouts (CFRs). Installation of optional updates isn't enabled by default for devices that receive updates using Windows Update client policies. However, you can enable optional updates for devices by using AllowOptionalContent. For more information about optional content, see Enable optional updates.

Feedback

Was this page helpful? Yes No

Provide product feedback

Windows subscription activation

Applies to: ✅ Windows 11, ✅ Windows 10

The subscription activation feature enables a "step-up" from Windows Pro edition to Enterprise edition or from Windows Pro Education edition to Education edition. This feature can be used with a subscription to Windows Enterprise E3 or E5 licenses.

> **Tip**
>
> Windows Pro Education is analogous to Windows Pro, while Windows Education is analogous to Windows Enterprise. In other words, Windows Education is a step-up from Windows Pro Education, similar to how Windows Enterprise is a step-up from Windows Pro.

The subscription activation feature eliminates the need to manually deploy Enterprise or Education edition images on each target device, then later:

- Standing up on-premises key management services such as KMS or MAK based activation.
- Entering Generic Volume License Keys (GVLKs).
- Rebooting client devices.

For more information on how to deploy Enterprise licenses, see Deploy Windows Enterprise licenses.

Subscription activation for Enterprise

Windows Enterprise E3 and E5 are available as online services via subscription. Windows Enterprise can be deployed in an organization without keys and reboots.

- Devices with a current Windows Pro edition license can be seamlessly upgraded to Windows Enterprise.
- Product key-based Windows Enterprise software licenses can be transitioned to Windows Enterprise subscriptions.

Organizations that have an enterprise agreement can also benefit from the service, using traditional Active Directory-joined devices. In this scenario, the Active Directory user that signs in on their device must be synchronized with Microsoft Entra ID using Microsoft Entra Connect Sync.

> **Note**

Subscription activation is available for qualifying devices running currently supported versions of Windows. Subscription activation can't be used to upgrade to a newer version of Windows.

Adding Conditional Access policy

Organizations that use the Subscription Activation feature to enable users to "step-up" from one version of Windows to another and use Conditional Access policies to control access need to exclude one of the following cloud apps from their Conditional Access policies using **Select Excluded Cloud Apps**:

- Universal Store Service APIs and Web Application, AppID 45a330b1-b1ec-4cc1-9161-9f03992aa49f.
- Windows Store for Business, AppID 45a330b1-b1ec-4cc1-9161-9f03992aa49f.

  > **Note**
  >
  > The Microsoft Store for Business and Microsoft Store for Education are retired. For more information, see Microsoft Store for Business and Education retiring March 31, 2023.

Although the app ID is the same in both instances, the name of the cloud app depends on the tenant.

For more information about configuring exclusions in Conditional Access policies, see Application exclusions.

When a device has been offline for an extended period of time, the device might not reactivate automatically if this Conditional Access exclusion isn't in place. Setting this Conditional Access exclusion ensures that Subscription Activation continues to work seamlessly.

Starting with Windows 11, version 23H2 with KB5034848 or later, users are prompted for authentication with a toast notification when Subscription Activation needs to reactivate. The toast notification will show the following message:

> **Your account requires authentication**
>
> **Please sign in to your work or school account to verify your information.**

Additionally, in the **Activation** pane, the following message might appear:

> **Please sign in to your work or school account to verify your information.**

The prompt for authentication usually occurs when a device has been offline for an extended period of time. This change eliminates the need for an exclusion in the Conditional Access policy for Windows 11, version 23H2 with KB5034848 or later. A Conditional Access policy can still be used with Windows 11, version 23H2 with KB5034848 or later if the prompt for user authentication via a toast notification isn't desired.

Subscription activation for Education

Subscription activation for Education works the same as the Enterprise edition. However, in order to use subscription activation for Education, the device must have Windows Pro Education and an active subscription plan with an Enterprise license. For more information, see the requirements section.

Inherited activation

Inherited activation allows Windows virtual machines to inherit activation state from their Windows client host. When a user with a Windows E3/E5 or A3/A5 license assigned creates a new Windows virtual machine (VM) using a Windows host, the VM inherits the activation state from a host machine. This behavior is independent of whether the user signs on with a local account or uses a Microsoft Entra account on a VM.

To support inherited activation, both the host computer and the VM must be running a currently supported version of Windows. The hypervisor platform must also be Windows Hyper-V.

Requirements

Windows Enterprise requirements

For Microsoft customers with Enterprise Agreements (EA) or Microsoft Products & Services Agreements (MPSA), the following requirements must be met:

- A supported version of Windows Pro or Enterprise edition installed on the devices to be upgraded.
- Microsoft Entra available for identity management.
- Devices must be Microsoft Entra joined or Microsoft Entra hybrid joined. Workgroup-joined or Microsoft Entra registered devices aren't supported.

For Microsoft customers that don't have EA or MPSA, Windows Enterprise E3/E5 or A3/A5 licenses can be obtained through a cloud solution provider (CSP). Identity management and device requirements are the same when using CSP to manage licenses. For more information about getting Windows Enterprise E3 through a CSP, see Windows Enterprise E3 in CSP.

> **Note**
>
> These requirements don't apply to general Windows client activation on Azure. Azure activation requires a connection to Azure KMS only. Azure KMS supports workgroup, hybrid, and Microsoft Entra joined VMs. In most scenarios, activation of Azure VMs happens automatically. For more information, see Understanding Azure KMS endpoints for Windows product activation of Azure virtual machines.

> **Important**
>
> As of October 1, 2022, subscription activation is available for *commercial* and *GCC* tenants. It's currently not available on GCC High or DoD tenants. For more information, see Enable subscription activation with an existing EA.

Windows Education requirements

- A supported version of Windows Pro Education installed on the devices to be upgraded.
- A device with a Windows Pro Education digital license. This information can be confirmed in the Activationpane of the **Settings** app under **Settings** > **System** > **Activation**.
- The Education tenant must have an active subscription to Microsoft 365 with a Windows Enterprise license, or a Windows Enterprise or Education subscription.
- Devices must be Microsoft Entra joined or Microsoft Entra hybrid joined. Workgroup-joined or Microsoft Entra registered devices aren't supported.

> **Important**
>
> If Windows Pro is converted to Windows Pro Education, then subscription activation doesn't work. The device needs to be reimaged to Windows Pro Education for subscription activation to work. Alternatively, reimage the device directly to Windows Education.

Benefits

With Windows Enterprise or Education editions, an organization can benefit from enterprise-level security and control. Previously, only organizations with a Microsoft Volume Licensing Agreement could deploy Education or Enterprise editions to their users. With Windows Enterprise E3/E5 or A3/A5 being available as an online service, it's available in select channels thus allowing all organizations to take advantage of enterprise-grade Windows features.

To compare Windows editions and review pricing, see the following sites:

- Compare Windows editions
- Enterprise Mobility + Security Pricing Options

Benefits of moving to Windows as an online service include:

- Licenses for Windows Enterprise and Education are checked based on Microsoft Entra credentials. There's a systematic way to assign licenses to end users and groups in an organization.
- User sign-in triggers a silent edition upgrade, with no reboot required.
- Support for mobile worker and "Bring Your Own Device" (BYOD) or "Choose Your Own Device" (CYOD) activation. This support transitions away from on-premises KMS and MAK keys.
- Compliance support via license assignment.
- Licenses can be updated to different users dynamically, which allows optimization of the licensing investment against changing needs.

How it works

The device is Microsoft Entra joined, for example from the Access work or school pane in the **Settings** app that can be found under **Settings** > **Accounts** > **Access work or school**.

Windows Enterprise is assigned to a user, for example through the Microsoft 365 admin center. When a licensed user signs in to a device that meets requirements using their Microsoft Entra credentials, Windows steps up from Pro edition to Enterprise, or from Pro Education to Education. Once the edition is stepped up, Enterprise/Education features are unlocked. When a user's subscription expires or is transferred to another user, the device reverts seamlessly to Windows Pro or Windows Pro Education edition, once the current subscription validity expires.

> **Note**

- Devices running a supported version of Windows Pro can get Windows Enterprise general availability channel on up to five devices for each user covered by the license. This limit also applies when stepping up from Windows Pro Education to Windows Education. This benefit doesn't include the long term servicing channel.
- A Windows Pro device only steps up to Windows Enterprise edition when a **Windows Enterprise** license is assigned from the Microsoft 365 admin center.
- A Windows Pro Education device only steps up to Windows Education edition a **Windows Enterprise** license is assigned from the Microsoft 365 admin center.

Scenarios

Scenario #1

A supported version of Windows is being used. A Windows Enterprise E3 or E5 subscription is purchased, or there's an existing E3 or E5 subscription but Windows Enterprise isn't yet deployed.

All of the Windows Pro devices step-up to Windows Enterprise. When a subscription activation-enabled user signs in, devices that are already running Windows Enterprise migrate from KMS or MAK activated Enterprise edition to subscription activated Enterprise edition.

Scenario #2

Microsoft Entra joined devices or Active Directory-joined devices running a supported version of Windows are being used. Microsoft Entra synchronization is configured. The steps in Deploy Windows Enterprise licenses are followed to get a $0 SKU and a new Windows Enterprise E3 or E5 license in Microsoft Entra ID. The license is then assigned to all of the Microsoft Entra users, which can be Active Directory-synced accounts. When that user signs in, the device automatically steps up from Windows Pro to Windows Enterprise or from Windows Pro Education to Windows Education.

Earlier versions of Windows

If devices are running Windows 7 or Windows 8.1, more steps are required. A wipe-and-load approach still works, but it can be easier to upgrade from Windows 7 Pro directly to a currently supported Windows 10 Enterprise edition. This path is supported, and completes the move in one step. However, versions of Windows newer than Windows 10 don't support upgrading

from Windows 7 or Windows 8.1. For versions of Windows newer than Windows 10, an upgrade to Windows 10 would first be required, followed by upgrading to the version of Windows Enterprise newer than Windows 10. In this scenario, a wipe-and-load might be more practical.

Licenses

The following policies apply to acquisition and renewal of licenses on devices:

- Upgraded devices attempt to renew licenses about every 30 days. They must be connected to the internet to successfully acquire or renew a license.
- If a device is disconnected from the internet, until its current subscription expires Windows reverts to Pro or Pro Education. As soon as the device is connected to the internet again, the license automatically renew.
- Up to five devices can be upgraded for each user license. If the user license is used for a sixth device, the computer where the user hasn't signed in for the longest time reverts to Pro or Pro Education.
- If a device meets the requirements and a licensed user signs in on that device, the device is upgraded.

Licenses can be reallocated from one user to another user, allowing optimization of the licensing investment against changing needs.

With a Microsoft Entra subscription, group-based licensing is the preferred method to assign Enterprise E3 and E5 licenses to users. For more information, see Group-based licensing basics in Microsoft Entra ID.

Existing Enterprise deployments

With currently supported versions of Windows, subscription activation automatically pulls the firmware-embedded Windows activation key and activates the underlying Pro license. The license then steps up to Enterprise using subscription activation. This behavior automatically migrates devices from KMS or MAK activated Enterprise to subscription activated Enterprise.

Subscription activation doesn't remove the need to activate the underlying OS. This requirement still exists for running a genuine installation of Windows.

Caution

Firmware-embedded Windows activation happens automatically only during Windows Setup out of box experience (OOBE).

If the computer has never been activated with a Pro key, use the following script from an elevated PowerShell console:

PowerShell

```
$(Get-WmiObject SoftwareLicensingService).OA3xOriginalProductKey | foreach{ if ( $null -ne $_ ) { Write-Host "Installing"$_;changepk.exe /Productkey $_ } else { Write-Host "No key present" } }
```

Obtaining a Microsoft Entra ID license

If an organization has an Enterprise Agreement (EA) or Software Assurance (SA):

- Organizations with a traditional EA must order a $0 SKU, process e-mails sent to the license administrator for the company, and assign licenses using Microsoft Entra ID. Ideally, licenses are assigned to groups using the Microsoft Entra ID P1 or P2 feature for group assignment. For more information, see Enable subscription activation with an existing EA.
- The license administrator can assign licenses to Microsoft Entra users with the same process used for Microsoft 365 Apps.
- New EA/SA Windows Enterprise customers can acquire both an SA subscription and an associated $0 cloud subscription.

If an organization has a Microsoft Products & Services Agreement (MPSA):

- New customers are automatically emailed the details of the service. Follow the instructions and steps found in the email.
- Existing MPSA customers receive service activation emails that allow their customer administrator to assign users to the service.
- New MPSA customers who purchase the Software Subscription Windows Enterprise E3 and E5 are enabled for both the traditional key-based and new subscriptions activation method.

Deploying licenses

For more information, see Deploy Windows Enterprise licenses.

Virtual Desktop Access (VDA)

Subscriptions to Windows Enterprise are also available for virtualized clients. Enterprise E3 and E5 are available for Virtual Desktop Access (VDA) in Microsoft Azure or in another qualified multitenant hoster (QMTH).

Virtual machines (VMs) must be configured to enable Windows Enterprise subscriptions for VDA. Active Directory-joined and Microsoft Entra joined clients are supported. See Enable VDA for Subscription Activation.

Related content

- Deploy Windows Enterprise licenses.
- Plan your Microsoft Entra hybrid join implementation.
- Compare Windows editions.
- Windows for business.

Last updated on 05/14/2025

Windows Enterprise E3 in CSP

Applies to: ✅ Windows 11, ✅ Windows 10

Windows Enterprise E3 in CSP delivers, by subscription, exclusive features reserved for Windows Enterprise editions. This offering is available through the Cloud Solution Provider (CSP) channel via the Partner Center as an online service. Windows Enterprise E3 in CSP provides a flexible, per-user subscription for small and medium-sized organizations (from one to hundreds of users). To take advantage of this offering, the following prerequisites must be met:

- A currently supported version of Windows, installed and activated, on the devices to be upgraded.
- Microsoft Entra available for identity management.

Moving from Windows Pro to Windows Enterprise is more easy than ever before with no keys and no reboots. After a user enters the Microsoft Entra credentials associated with a Windows Enterprise E3 license, the operating system turns from Windows Pro to Windows Enterprise, and all the appropriate Enterprise features are unlocked. When a subscription license expires or is transferred to another user, the Enterprise device seamlessly steps back down to Windows Pro.

Previously, only organizations with a Microsoft Volume Licensing Agreement could deploy Windows Enterprise to their users. Now, with Windows Enterprise E3 in CSP, small- and medium-sized organizations can more easily take advantage of Enterprise edition features.

When Windows Enterprise E3 is purchased via a partner, the following benefits are included:

- **Windows Enterprise edition**. Devices currently running Windows Pro can get Windows Enterprise Current Branch (CB) or Current Branch for Business (CBB). This benefit doesn't include Long Term Service Branch (LTSB).
- **Support from one to hundreds of users**. Although the Windows Enterprise E3 in CSP program doesn't have a limitation on the number of licenses an organization can have, the program is designed for small- and medium-sized organizations.
- **Deploy on up to five devices**. For each user covered by the license, Windows Enterprise edition can be deployed on up to five devices.
- **Roll back to Windows Pro at any time**. When a user's subscription expires or is transferred to another user, the Windows Enterprise device reverts seamlessly to Windows Pro edition (after a grace period of up to 90 days).
- **Monthly, per-user pricing model**. This model makes Windows Enterprise E3 affordable for organizations.
- **Move licenses between users**. Licenses can be quickly and easily reallocated from one user to another user, allowing optimization of the licensing investment against changing

needs.

How does the Windows Enterprise E3 in CSP program compare with Microsoft Volume Licensing Agreements and Software Assurance?

- Microsoft Volume Licensing programs are broader in scope, providing organizations with access to licensing for all Microsoft products.
- Software Assurance provides organizations with the following categories of benefits:
  - **Deployment and management**. These benefits include planning services:
    - Microsoft Desktop Optimization (MDOP).
    - Windows Virtual Desktop Access Rights.
    - Windows Roaming Use Rights.
    - Other benefits.
  - **Training**. These benefits include training vouchers, online e-learning, and a home use program.
  - **Support**. These benefits include:
    - 24x7 problem resolution support.
    - Backup capabilities for disaster recovery.
    - System Center Global Service Monitor.
    - A passive secondary instance of SQL Server.
  - **Specialized**. These benefits include step-up licensing availability, which enables migration of software from an earlier edition to a higher-level edition. It also spreads license and Software Assurance payments across three equal, annual sums.

    In addition, in Windows Enterprise E3 in CSP, a partner can manage the licenses for an organization. With Software Assurance, the organization has to manager their own licenses.

In summary, the Windows Enterprise E3 in CSP program is an upgrade offering that provides small- and medium-sized organizations easier, more flexible access to the benefits of Windows Enterprise edition. Microsoft Volume Licensing programs and Software Assurance on the other hand are broader in scope and provide benefits beyond access to the Enterprise edition of Windows.

Compare Windows Pro and Enterprise editions

Windows Enterprise edition has many features that are unavailable in Windows Pro. Table 1 lists some of the Windows Enterprise features not found in Windows Pro. Many of these features are security-related, whereas others enable finer-grained device management.

Table 1. Windows Enterprise features not found in Windows Pro

Expand table

Feature	Description
Credential Guard	Credential Guard uses virtualization-based security to help protect security secrets so that only privileged system software can access them. Examples of security secrets that can be protected include NTLM password hashes and Kerberos Ticket Granting Tickets. This protection helps prevent Pass-the-Hash or Pass-the-Ticket attacks. Credential Guard has the following features: • **Hardware-level security** - Credential Guard uses hardware platform security features (such as Secure Boot and virtualization) to help protect derived domain credentials and other secrets. • **Virtualization-based security** - Windows services that access derived domain credentials and other secrets run in a virtualized, protected environment that is isolated. • **Improved protection against persistent threats** - Credential Guard works with other technologies (for example, Device Guard) to help provide further protection against attacks, no matter how persistent. • **Improved manageability** - Credential Guard can be managed through Group Policy, Windows Management Instrumentation (WMI), or Windows PowerShell. For more information, see Protect derived domain credentials with Credential Guard. *Credential Guard requires* *UEFI 2.3.1 or greater with Trusted Boot* *Virtualization Extensions such as Intel VT-x, AMD-V, and SLAT must be enabled* *x64 version of Windows* *IOMMU, such as Intel VT-d, AMD-Vi* *BIOS Lockdown* • *TPM 2.0 recommended for device health attestation (uses software if TPM 2.0 not present)*
AppLocker management	This feature helps IT pros determine which applications and files users can run on a device. The applications and files that can be managed include executable files, scripts, Windows Installer files, dynamic-link libraries (DLLs), packaged apps, and packaged app installers. For more information, see AppLocker.
Application Virtualization (App-V)	This feature makes applications available to end users without installing the applications directly on users' devices. App-V transforms applications into centrally managed services that are never installed and don't conflict with other applications. This feature also helps ensure that applications are kept current with the latest

Feature	Description
	security updates. For more information, see Getting started with App-V for Windows client.
User Experience Virtualization (UE-V)	With this feature, user-customized Windows and application settings can be captured and stored on a centrally managed network file share. When users sign in, their personalized settings are applied to their work session, regardless of which device or virtual desktop infrastructure (VDI) sessions they sign into. UE-V provides the following features: • Specify which application and Windows settings synchronize across user devices • Deliver the settings anytime and anywhere users work throughout the enterprise • Create custom templates for line-of-business applications • Recover settings after hardware replacement or upgrade, or after reimaging a virtual machine to its initial state For more information, see User Experience Virtualization (UE-V) overview.
Managed User Experience	This feature helps customize and lock down a Windows device's user interface to restrict it to a specific task. For example, a device can be configured for a controlled scenario such as a kiosk or classroom device. The user experience would be automatically reset once a user signs off. Access to services such as the Windows Store can also be restricted. For Windows 10, Start layout options can also be managed, such as: • Removing and preventing access to the Shut Down, Restart, Sleep, and Hibernate commands • Removing Log Off (the User tile) from the Start menu • Removing frequent programs from the Start menu • Removing the All Programs list from the Start menu • Preventing users from customizing their Start screen • Forcing Start menu to be either full-screen size or menu size • Preventing changes to Taskbar and Start menu settings

Deployment of Windows Enterprise E3 licenses

See Deploy Windows Enterprise licenses.

Deploy Windows Enterprise features

Now that Windows Enterprise edition is running on devices, how are Enterprise edition features and capabilities taken advantage of? What are the next steps that need to be taken for each of the features discussed in Table 1?

The following sections provide with the high-level tasks that need to be performed in an environment to help users take advantage of the Windows Enterprise edition features.

Credential Guard

> **Note**
>
> Requires UEFI 2.3.1 or greater with Trusted Boot; Virtualization Extensions such as Intel VT-x, AMD-V, and SLAT must be enabled; x64 version of Windows; IOMMU, such as Intel VT-d, AMD-Vi; BIOS Lockdown; TPM 2.0 recommended for device health attestation (will use software if TPM 2.0 not present).

Credential Guard can be implemented on Windows Enterprise devices by turning on Credential Guard on these devices. Credential Guard uses Windows virtualization-based (Hyper-V) security features that must be enabled on each device before Credential Guard can be turned on. Credential Guard can be turned on by using one of the following methods:

- **Automated**. Credential Guard can be turned on for one or more devices by using Group Policy. The Group Policy settings automatically add the virtualization-based security features and configure the Credential Guard registry settings on managed devices.
- **Manual**. Credential Guard can be manually turned on by taking one of the following actions:
  - Add the virtualization-based security features by using Programs and Features or Deployment Image Servicing and Management (DISM).
  - Configure Credential Guard registry settings by using the Registry Editor or the Device Guard and Credential Guard hardware readiness tool .

    These manual steps can be automated by using a management tool such as Microsoft Configuration Manager.

For more information about implementing Credential Guard, see the following resources:

- Credential Guard overview
- Security considerations for Original Equipment Manufacturers
- Device Guard and Credential Guard hardware readiness tool

AppLocker management

AppLocker in Windows Enterprise can be managed by using Group Policy. Group Policy requires having AD DS and that the Windows Enterprise devices are joined to an AD DS domain. AppLocker rules can be created by using Group Policy. The AppLocker rules can then be targeted to the appropriate devices.

For more information about AppLocker management by using Group Policy, see AppLocker deployment guide.

App-V

App-V requires an App-V server infrastructure to support App-V clients. The primary App-V components that are required are:

- **App-V server**. The App-V server provides App-V management, virtualized app publishing, app streaming, and reporting services. Each of these services can be run on one server or can be run individually on multiple servers. For example, multiple streaming servers might exist. App-V clients contact App-V servers to determine which apps are published to the user or device, and then run the virtualized app from the server.

- **App-V sequencer**. The App-V sequencer is a typical client device that is used to sequence (capture) apps and prepare them for hosting from the App-V server. Apps are installed on the App-V sequencer, and the App-V sequencer software determines the files and registry settings that are changed during app installation. Then the sequencer captures these settings to create a virtualized app.

- **App-V client**. The App-V client must be enabled on any Windows Enterprise E3 client device that needs to run apps from the App-V server.

For more information about implementing the App-V server, App-V sequencer, and App-V client, see the following resources:

- Getting started with App-V for Windows client
- Deploying the App-V server
- Deploying the App-V Sequencer and Configuring the Client

UE-V

UE-V requires server and client-side components that need to be downloaded, activated, and installed. These components include:

- **UE-V service**. The UE-V service (when enabled on devices) monitors registered applications and Windows for any settings changes, then synchronizes those settings between devices.

- **Settings packages**. Settings packages created by the UE-V service store application settings and Windows settings. Settings packages are built, locally stored, and copied to the settings storage location.

- **Settings storage location**. This location is a standard network share that users can access. The UE-V service verifies the location and creates a hidden system folder in which to store and retrieve user settings.

- **Settings location templates**. Settings location templates are XML files that UE-V uses to monitor and synchronize desktop application settings and Windows desktop settings between user computers. By default, some settings location templates are included in UE-V. Custom settings location templates can also be created, edited, or validated by using the UE-V template generator. Settings location templates aren't required for Windows applications.

- **Universal Windows applications list**. UE-V determines which Windows applications are enabled for settings synchronization using a managed list of applications. By default, this list includes most Windows applications.

For more information about deploying UE-V, see the following resources:

- User Experience Virtualization (UE-V) overview
- Get Started with UE-V
- Prepare a UE-V Deployment

Managed User Experience

The Managed User Experience feature is a set of Windows Enterprise edition features and corresponding settings that can be used to manage user experience. Table 2 describes the Managed User Experience settings (by category), which are only available in Windows Enterprise edition. The management methods used to configure each feature depend on the feature. Some features are configured by using Group Policy, while others are configured by using Windows PowerShell, Deployment Image Servicing and Management (DISM), or other command-line tools. For the Group Policy settings, AD DS is required with the Windows Enterprise devices joined to an AD DS domain.

Table 2. Managed User Experience features

Expand table

Feature	Description
Start layout customization	A customized Start layout can be deployed to users in a domain. No reimaging is required, and the Start layout can be updated simply by overwriting the .xml file that contains the layout. The XML file enables customization of Start layouts for different departments or organizations, with minimal management overhead. For more information on these settings, see Customize Windows 10 Start and taskbar with Group Policy.
Unbranded boot	Windows elements that appear when Windows starts or resumes can be suppressed. The crash screen when Windows encounters an error from which it can't recover can also be suppressed. For more information on these settings, see Unbranded Boot.
Custom Logon	The Custom Logon feature can be used to suppress Windows UI elements that relate to the Welcome screen and shutdown screen. For example, all elements of the Welcome screen UI can be suppressed and a custom logon UI can be provided. The Blocked Shutdown Resolver (BSDR) screen can also be suppressed and applications can be automatically ended while the OS waits for applications to close before a shutdown. For more information on these settings, see Custom Logon.
Shell launcher	Enables Assigned Access to run only a classic Windows app via Shell Launcher to replace the shell. For more information on these settings, see Shell Launcher.
Keyboard filter	Keyboard Filter can be used to suppress undesirable key presses or key combinations. Normally, users can use certain Windows key combinations like Ctrl+Alt+Delete or Ctrl+Shift+Tab to control a device by locking the screen or using Task Manager to close a running application. These keyboard actions aren't desirable on devices intended for a dedicated purpose. For more information on these settings, see Keyboard Filter.
Unified write filter	The Unified Write Filter (UWF) can be used on a device to help protect physical storage media, including most standard writable storage types supported by Windows, such as: • Physical hard disks • Solid-state drives • Internal USB devices • External SATA devices . UWF can also be used to make read-only media appear to the OS as a writable volume. For more information on these settings, see Unified Write Filter.

Related articles

- Windows Enterprise Subscription Activation.
- Plan your Microsoft Entra hybrid join implementation.
- Compare Windows editions.

- Windows for business .

Last updated on 05/14/2025

Configure VDA for Windows subscription activation

Applies to:

- Windows 10
- Windows 11

This document describes how to configure virtual machines (VMs) to enable Windows subscription activation in a Windows Virtual Desktop Access (VDA) scenario. Windows VDA is a device or user-based licensing mechanism for managing access to virtual desktops.

Deployment instructions are provided for the following scenarios:

1. Active Directory-joined VMs
2. Microsoft Entra joined VMs
3. Azure Gallery VMs

Requirements

- VMs must be running a supported version of Windows Pro edition.
- VMs must be joined to Active Directory or Microsoft Entra ID.
- VMs must be hosted by a Qualified Multitenant Hoster (QMTH).

Activation

Scenario 1

- The VM is running a supported version of Windows.

- The VM is hosted in Azure, an authorized outsourcer, or another Qualified Multitenant Hoster (QMTH).

  When a user with VDA rights signs in to the VM using their Microsoft Entra credentials, the VM is automatically stepped-up to Enterprise and activated. There's no need to do Windows Pro activation. This functionality eliminates the need to maintain KMS or MAK in the qualifying cloud infrastructure.

Scenario 2

- The Hyper-V host and the VM are both running a supported version of Windows.

Inherited Activation is enabled. All VMs created by a user with a Windows E3 or E5 license are automatically activated independent of whether a user signs in with a local account or using a Microsoft Entra account.

Scenario 3

- The hoster isn't an authorized QMTH partner.

  In this scenario, the underlying Windows Pro license must be activated prior to using subscription activation Windows Enterprise. Activation is accomplished using a generic volume license key (GVLK) and a volume license KMS activation server provided by the hoster. Alternatively, a KMS activation server can be used. KMS activation is provided for Azure VMs. For more information, see Troubleshoot Azure Windows virtual machine activation problems.

For examples of activation issues, see Troubleshoot the user experience.

Active Directory-joined VMs

1. Use the following instructions to prepare the VM for Azure: Prepare a Windows VHD or VHDX to upload to Azure

2. (Optional) To disable network level authentication, enter the following command at an elevated command prompt:

   Windows Command Prompt

   ```
   REG.exe ADD "HKLM\SYSTEM\CurrentControlSet\Control\Terminal
   Server\WinStations\RDP-Tcp" /v UserAuthentication /t REG_DWORD /d 0 /f
   ```

3. At an elevated command prompt, enter **sysdm.cpl**.

4. On the Remote tab, choose **Allow remote connections to this computer** and then select **Select Users**.

5. Select **Add**, enter **Authenticated users**, and then select **OK** three times.

6. Follow the instructions to use sysprep at Steps to generalize a VHD and then start the VM again.

7. If you must activate Windows Pro as described for scenario 3, complete the following steps to use Windows Configuration Designer and inject an activation key. Otherwise, skip to step 8.

a. Install Windows Configuration Designer.

b. Open Windows Configuration Designer and select **Provision desktop services**.

c. Under **Name**, enter **Desktop AD Enrollment Pro GVLK**, select **Finish**, and then on the **Set up device** page enter a device name.

> ⓘ **Note**
>
> You can use a different project name, but this name is also used with dism.exe in a later step.

d. Under **Enter product key** enter the Pro GVLK key: `W269N-WFGWX-YVC9B-4J6C9-T83GX`.

e. On the Set up network page, choose **Off**.

f. On the Account Management page, choose **Enroll into Active Directory** and then enter the account details.

> ⓘ **Note**
>
> This step is different for Microsoft Entra joined VMs.

g. On the Add applications page, add applications if desired. This step is optional.

h. On the Add certificates page, add certificates if desired. This step is optional.

i. On the Finish page, select **Create**.

j. In file explorer, open the VHD to mount the disk image. Determine the drive letter of the mounted image.

k. Enter the following command at an elevated command prompt. Replace the letter `G` with the drive letter of the mounted image, and enter the project name you used if it's different than the one suggested:

Windows Command Prompt

```
Dism.exe /Image=G:\ /Add-ProvisioningPackage /PackagePath: "Desktop AD
Enrollment Pro GVLK.ppkg"
```

l. Right-click the mounted image in file explorer and select **Eject**.

8. See the instructions at Upload and create VM from generalized VHD to sign in to Azure, get your storage account details, upload the VHD, and create a managed image.

Microsoft Entra joined VMs

> **Important**
>
> Microsoft Entra provisioning packages have a 180 day limit on bulk token usage. After 180 days, you'll need to update the provisioning package and re-inject it into the image. Existing virtual machines that are Microsoft Entra joined and deployed won't need to be recreated.

For Microsoft Entra joined VMs, follow the same instructions as for Active Directory-joined VMs with the following exceptions:

- During setup with Windows Configuration Designer, under **Name**, enter a name for the project that indicates it isn't for Active Directory-joined VMs, such as **Desktop Bulk Enrollment Token Pro GVLK**.
- During setup with Windows Configuration Designer, on the Account Management page, instead of enrolling in Active Directory, choose **Enroll in Microsoft Entra ID**, select **Get Bulk Token**, sign in, and add the bulk token using your organization's credentials.
- When entering the PackagePath, use the project name you previously entered. For example, **Desktop Bulk Enrollment Token Pro GVLK.ppkg**
- When attempting to access the VM using remote desktop, you'll need to create a custom RDP settings file as described below in Create custom RDP settings for Azure.

Azure Gallery VMs

1. (Optional) To disable network level authentication, enter the following command at an elevated command prompt:

   Windows Command Prompt

   ```
   REG.exe ADD "HKLM\SYSTEM\CurrentControlSet\Control\Terminal
   Server\WinStations\RDP-Tcp" /v UserAuthentication /t REG_DWORD /d 0 /f
   ```

2. At an elevated command prompt, enter `sysdm.cpl`.

3. On the Remote tab, choose **Allow remote connections to this computer** and then select **Select Users**.

4. Select **Add**, enter **Authenticated users**, and then select **OK** three times.

5. Install Windows Configuration Designer.

6. Open Windows Configuration Designer and select **Provision desktop services**.

7. If you must activate Windows Pro as described for scenario 3, complete the following steps. Otherwise, skip to step 8.

   a. Under **Name**, enter **Desktop Bulk Enrollment Token Pro GVLK**, select **Finish**, and then on the **Set up device** page enter a device name.

   b. Under **Enter product key** enter the Pro GVLK key: `W269N-WFGWX-YVC9B-4J6C9-T83GX`.

8. Under **Name**, enter **Desktop Bulk Enrollment**, select **Finish**, and then on the **Set up device** page enter a device name.

9. On the Set up network page, choose **Off**.

10. On the Account Management page, choose **Enroll in Microsoft Entra ID**, select **Get Bulk Token**, sign in, and add the bulk token using your organizations credentials.

11. On the Add applications page, add applications if desired. This step is optional.

12. On the Add certificates page, add certificates if desired. This step is optional.

13. On the Finish page, select **Create**.

14. Copy the PPKG file to the remote virtual machine. Open the provisioning package to install it. This process will restart the system.

> ⓘ **Note**
>
> When you try to access the VM using remote desktop, you'll need to create a custom RDP settings file.

Create custom RDP settings for Azure

1. Open Remote Desktop Connection and enter the IP address or DNS name for the remote host.

2. Select **Show Options**, and then under Connection settings select **Save As**. Save the RDP file to the location where you'll use it.

3. Close the Remote Desktop Connection window and open Notepad.

4. Open the RDP file in Notepad to edit it.

5. Enter or replace the line that specifies authentication level with the following two lines of text:

   ```text
   enablecredsspsupport:i:0
   authentication level:i:2
   ```

   The values `enablecredsspsupport` and `authentication level` should each appear only once in the file.

6. Save your changes, and then use this custom RDP file with your Microsoft Entra credentials to connect to the Azure VM.

Related articles

Windows subscription activation

Recommended settings for VDI desktops

Whitepaper on licensing the Windows desktop for VDI environments

---

Last updated on 05/14/2025

Deploy Windows Enterprise licenses

Applies to: ✅ Windows 11, ✅ Windows 10

This article describes how to deploy Windows Enterprise E3 or E5 licenses with subscription activation or Enterprise E3 in CSP and Microsoft Entra ID.

These activation features require a supported and licensed version of Windows Pro:

- Subscription activation with an enterprise agreement (EA) or a Microsoft Products & Services Agreement (MPSA).
- Enterprise E3 in CSP.
- Automatic, non-KMS activation also requires a device with a firmware-embedded activation key.
- Subscription activation requires Enterprise *per user* licensing. It doesn't work with *per device* licensing.

Enable subscription activation with an existing EA

EA customers with an existing Microsoft 365 tenant can use the following steps to enable Windows subscription licenses on the existing tenant:

1. Work with the reseller to place an order for one $0 SKU per user. As of October 1, 2022, there are three SKUs available, depending on the current Windows Enterprise SA license:

   Expand table

SKU	Description
AAA-51069	`Win OLS Activation User Alng Sub Add-on E3`
AAA-51068	`Win OLS Activation User Sub Add-on E5`
VRM-00001	`Win OLS Activation User GCC Sub Per User`

   > **Note**
   >
   > As of October 1, 2022, subscription activation is available for *commercial* and *GCC* tenants. It's currently not available on GCC High or DoD tenants.

2. After an order is placed, the OLS admin on the agreement will receive a service activation email, which indicates the subscription licenses is provisioned on the tenant.

3. Subscription licenses can now be assigned to users.

To update contact information and resend the activation email, use the following process:

1. Sign in to the Microsoft Volume Licensing Service Center.
2. Select **Subscriptions**.
3. Select **Online Services Agreement List**.
4. Enter the agreement number, and then select **Search**.
5. Select the **Service Name**.
6. In the **Subscription Contact** section, select the name listed under **Last Name**.
7. Update the contact information, then select **Update Contact Details**. This action triggers a new email.

Prepare for deployment: reviewing requirements

- Devices must be running a supported version of Windows Pro.
- Microsoft Entra joined, or hybrid domain joined with Microsoft Entra Connect. Customers who are federated with Microsoft Entra ID are also eligible.

For more information, see Review requirements on devices, later in this article.

Active Directory synchronization with Microsoft Entra ID

If there's an on-premises Active Directory Domain Services (AD DS) domain, identities in the on-premises AD DS domain need to be synchronized with Microsoft Entra ID. This synchronization is required for users to have a *single identity* that they can use to access their on-premises apps and cloud services that use Microsoft Entra ID. An example of a cloud service is Windows Enterprise E3 or E5.

For more information about integrating on-premises AD DS domains with Microsoft Entra ID, see the following resources:

- Configure Microsoft Entra hybrid join
- What is hybrid identity with Microsoft Entra ID?
- Microsoft Entra Connect and Microsoft Entra Connect Health installation roadmap

Assign licenses to users

After the Windows subscription is ordered, an email is sent with guidance on how to use Windows as an online service. The following methods are available to assign licenses:

- When the required Microsoft Entra subscription is available, group-based licensing is the preferred method to assign Enterprise E3 or E5 licenses to users.
- Licenses can be manually assigned by signing into the Microsoft 365 admin center.
- Licenses can be assigned by uploading a spreadsheet.
- Licenses can be assigned via PowerShell.

Explore the upgrade experience

Now that a subscription is established and licenses are assigned to users, devices running supported versions of Windows Pro can be upgraded to Enterprise edition.

> **Tip**
>
> This upgrade experience walkthrough assumes Autopilot isn't being used. For the Autopilot experience when joining Microsoft Entra ID, see User-driven Microsoft Entra join: Deploy the device.

Step 1: Join Windows Pro devices to Microsoft Entra ID

The first time the device starts, a Windows Pro device can join Microsoft Entra ID during setup. Existing devices can also join Microsoft Entra ID.

Join a device to Microsoft Entra ID during OOBE when the device is started for the first time

1. Power on the device for the first time to initiate Windows Setup and the Out of Box experience (OOBE).
2. In the **Is this the right country or region?** screen, select the desired country/region and then select the **Yes** button.
3. In the **Is this the right keyboard layout or input method?** screen, select the desired keyboard/input methods and then select the **Yes** button.
4. In the **Want to add a second keyboard layout?** screen, if desired add additional keyboard/input methods by selecting **Add layout**. Otherwise select the **Skip** button.

5. If no network connection is detected, the **Let's connect you to a network** screen appears. Connect to a wireless or wired network that has Internet access, and then select the **Next** button.

6. At this point, updates for Windows Setup might be installed. If updates are installed, the device reboots to finish installing the updates.

7. In Windows 11 Pro editions, the **Let's name your device** screen appears. Give the device a name and then select the **Next** button. After the device is given a name, the device might reboot.

8. In Windows 11 Pro editions, the **How would you like to set up this device?** screen appears. Select **Set up for work or school** and then select the **Next** button.

9. In the **Let's set things up for your work or school** screen:

   a. In the **someone@example.com** text box under **Sign in**, enter the username for the Microsoft Entra user account, and then select the **Next** button. The username is in the email format of user@domain.com.

   b. In the **Password** text box under **Enter password**, enter the password for the Microsoft Entra user account, and then select the **Sign in** button.

10. The device proceeds with the rest of the Windows setup including configuration of organization specific settings.

11. In the **Choose privacy settings for your device** screen, configure privacy settings as desired, using the **Next** button to go between settings. Once complete, select the **Accept** button.

12. Depending on the device and the configuration of organization specific settings, additional screens might appear. For example, the **Windows Hello** screen might appear.

Once Windows Setup finishes, the user is automatically signed in and the device is Microsoft Entra joined to the organization's subscription.

Join a device to Microsoft Entra ID when the device is already set up with Windows

> **Important**
>
> Make sure that the user signing in isn't the **BUILTIN/Administrator** account. That user can't use the `+ Connect` action to join a work or school account.

Open the **Accounts** > **Access work or school** pane in the **Settings** app by selecting the following link:

Access work or school

or

1. Right-click on the **Start** menu and select **Run**.
2. In the **Run** window, next to **Open:**, enter:

   Console
   ```
   ms-settings:workplace
   ```

   and then select **OK**.

or

1. Right-click on the **Start** menu and select **Settings**.
2. In the **Settings** app, select **Accounts** in the left hand pane.
3. In the **Accounts** pane, select **Access work or school**.

Once the **Accounts** > **Access work or school** pane is open:

1. In the **Accounts** > **Access work or school** pane, next to **Add a work or school account**, select the **Connect** button.
2. In the **Microsoft account** window that opens:
   a. In the **Set up a work or school account** page, under **Alternate actions:**, select **Join this device to Microsoft Entra ID**.
   b. In the **Email or phone** text box of the **Sign in** page, enter the username for the Microsoft Entra user account, and then select the **Next** button. The username is in the email format of user@domain.com.
   c. In the **Password** text box of the **Enter password** page, enter the password for the Microsoft Entra user account, and then select the **Sign in** button.
   d. When the **Make sure this is your organization** window opens, confirm the information is correct and then select the **Join** button.
   e. The device joins the organization's Microsoft Entra ID subscription. Once complete, the **You're all set!** page is displayed. Select the **Done** button to complete the process.

The device is now Microsoft Entra joined to the organization's subscription.

Step 2: Pro edition activation

Windows Pro has to be activated on the device. However, if the device is running a currently supported version of Windows, most modern devices automatically activates Windows Pro edition using the firmware-embedded activation key.

Step 3: Sign in using Microsoft Entra account

Once the device is joined to Microsoft Entra ID and Windows Setup/OOBE completes, the user signs in with their Microsoft Entra account. Once the user signs in with their Microsoft Entra account, the Windows Enterprise E3 or E5 license associated with the user enables Windows Enterprise edition capabilities on the device.

Step 4: Verify that Enterprise edition is enabled

To verify the Windows Enterprise E3 or E5 subscription:

Open the **Activation** pane in the **Settings** app by selecting the following link:

Activation

or

1. Right-click on the **Start** menu and select **Run**.
2. In the **Run** window, next to **Open:**, enter:

   Console
   ```
   ms-settings:activation
   ```

   and then select **OK**.

or

1. Right-click on the **Start** menu and select **Settings**.
2. In the **Settings** app, select **System** in the left hand pane.
3. In the **System** pane, **Activation**.

Once the **System** > **Activation** pane is open:

1. In the **System** > **Activation** pane, expand **Activation state** and **Subscription** to see full details of the activation state and status:

   a. Under **Activation state**, verify that Windows is activated. It should display the message:

   `Windows is activated with a digital license`

   b. Under **Subscription**, verify that the Windows 11 Enterprise subscription is active. It should display the message:

   `Windows 11 Enterprise subscription is active`

   > **Note**
   >
   > If the Windows Enterprise subscription hasn't yet been applied, the **Subscription** pane isn't displayed.

A device is healthy when both the subscription and activation are active. If there are any problems with the Windows Enterprise E3 or E5 license or the activation of the license, the **Activation** pane displays the appropriate error message or status. This information can be used to help diagnose the licensing and activation process.

Verify that Enterprise edition is enabled with slmgr

Slmgr can also be used to verify the activation information:

1. Open a command prompt.

2. To get basic licensing information, run the following command at the command prompt:

   Windows Command Prompt

   ```
   slmgr /dli
   ```

   A window with output similar to the following opens:

   Console

   ```
   Name: Windows(R), Professional edition
   Description: Windows(R) Operating System, RETAIL channel
   Partial Product Key: 3V66T
   License Status: Licensed
   ```

To instead get detailed licensing information, run the following command:

Windows Command Prompt

```
slmgr /dlv
```

For more information on **Slmgr**, see Slmgr.vbs options for obtaining volume activation information.

Troubleshoot the user experience

In some instances, users might experience problems with activation of the Windows Enterprise E3 or E5 subscription. The most common problems that users might experience are the following issues:

- The Windows Enterprise E3 or E5 subscription has lapsed, was removed, or isn't applied.
- Windows Pro was never activated.

When there are problems with Windows Enterprise E3 or E5 subscription activation, the following are errors can occur in the **Activation** pane:

- **Windows Pro isn't activated**

  When Windows Pro isn't activated on a device, the following message is displayed for **Activation** in the **Activation** pane:

  `Windows is not activated`

  Additionally, the following message might be displayed:

  `We can't activate Windows on this device right now. You can try activating again later or go to the Store to buy genuine Windows. Error code: 0xC004F034.`

  Examples where this problem can occur include:
  - The device doesn't have a firmware-embedded activation key.
  - The starting edition of Windows wasn't Windows Pro. For example, the starting edition of Windows was Windows Home.

  In these cases, a Windows Pro key might need to be manually entered.

- **Windows Enterprise subscription isn't active**

  When a device with a Windows Enterprise subscription has lapsed or has been removed, the following message is displayed for **Subscription** in the **Activation** pane:

  `Windows Enterprise subscription isn't valid.`

> **Note**
>
> If the Windows Enterprise subscription has never been applied, the **Subscription** pane isn't displayed.

Review requirements on devices

When there are Windows Enterprise E3 or E5 license activation issues on a device, verify that it meets all of the requirements:

- Devices must be running a currently supported version of Windows Pro. Versions of Windows Pro that are out support don't support this feature.
- Devices must be joined to Microsoft Entra ID, or hybrid domain joined with Microsoft Entra Connect. Customers who are federated with Microsoft Entra ID are also eligible.
- For automatic activation of Windows Pro, the device must have a firmware-embedded activation key.

Use the following guides to verify each one of these requirements:

- **Determine if the version of Windows is currently supported**.

  To determine if the version of Windows is currently supported:

  1. Open a command prompt
  2. In the command prompt window, enter:

     Windows Command Prompt

     ```
     winver.exe
     ```

  3. The **About Windows** window opens and displays both the OS version and the build information of Windows.
  4. Compare the information from the **About Windows** window against the Windows support lifecycle:
     - Windows 11 release information.
     - Windows 10 release information.

- **Determine if a device is Microsoft Entra joined**.

  To determine if a device is Microsoft Entra joined:

1. Open a command prompt.

2. In the command prompt window, enter:

   Windows Command Prompt

   ```
   dsregcmd.exe /status
   ```

3. Review the output. Under the first section called **Device State**, verify that the value of **AzureAdJoined** is **YES**. If the value is **YES**, the device is joined to Microsoft Entra ID.

   Console

   ```
   +----------------------------------------------------------------------+
   | Device State                                                         |
   +----------------------------------------------------------------------+

            AzureAdJoined : YES
         EnterpriseJoined : NO
             DomainJoined : NO
          Virtual Desktop : NOT SET
              Device Name : Demo-PC
   ```

- **Determine if devices has a firmware-embedded activation key**.

  To determine if the device has a firmware-embedded activation key:

  1. Open an elevated Windows PowerShell command prompt.

  2. In the elevated Windows PowerShell command prompt, enter:

     PowerShell

     ```
     (Get-CimInstance -query 'select * from
     SoftwareLicensingService').OA3xOriginalProductKey
     ```

  3. If the device has a firmware-embedded activation key, the key is displayed in the output. If the output is blank, the device doesn't have a firmware embedded activation key. Most modern OEM-provided devices designed to run currently supported versions of Windows have a firmware-embedded key.

- **Make sure the Microsoft Entra user has been assigned a license**.

  For more information, see Assigning licenses to users.

Recommended practices

Adding Conditional Access policy

When a device has been offline for an extended period of time, the Subscription Activation might not reactivate automatically on the device. To resolve this issue, use Conditional Access policies to control access by excluding one of the following cloud apps from their Conditional Access policies using **Select Excluded Cloud Apps**:

- Universal Store Service APIs and Web Application, AppID 45a330b1-b1ec-4cc1-9161-9f03992aa49f.
- Windows Store for Business, AppID 45a330b1-b1ec-4cc1-9161-9f03992aa49f.

> **Note**
>
> The Microsoft Store for Business and Microsoft Store for Education are retired. For more information, see Microsoft Store for Business and Education retiring March 31, 2023.

Although the app ID is the same in both instances, the name of the cloud app depends on the tenant.

For more information about configuring exclusions in Conditional Access policies, see Application exclusions.

Setting this Conditional Access policy ensures that Subscription Activation continues to work seamlessly.

Starting with Windows 11, version 23H2 with KB5034848 or later, users are prompted for authentication with a toast notification when Subscription Activation needs to reactivate. The toast notification will show the following message:

> **Your account requires authentication**
>
> **Please sign in to your work or school account to verify your information.**

Additionally, in the **Activation** pane, the following message might appear:

> **Please sign in to your work or school account to verify your information.**

The prompt for authentication usually occurs when a device has been offline for an extended period of time. This change eliminates the need for an exclusion in the Conditional Access policy for Windows 11, version 23H2 with KB5034848 or later. A Conditional Access policy can still be used with Windows 11, version 23H2 with KB5034848 or later if the prompt for user authentication via a toast notification isn't desired.

Make sure Windows Update isn't blocked

If a device isn't able to connect to Windows Update, it can lose activation status or be blocked from upgrading to Windows Enterprise. Make sure that Windows Update isn't blocked on the device:

- Using `gpedit.msc` or group policy editor in the domain, make sure that the following group policy setting is set to **Disabled** or **Not Configured**:

  Computer Configuration > **Administrative Templates** > **Windows Components** > **Windows Update** > **Manage updates offered from Windows Server Update Service** > **Do not connect to any Windows Update Internet locations**

  If this policy is set to **Enabled**, it must be changed to **Disabled** or **Not Configured**.

- In the following registry key of the registry:

  `HKLM\SOFTWARE\Policies\Microsoft\Windows\WindowsUpdate`

  check if the value `DoNotConnectToWindowsUpdateInternetLocations` exists. If the value does exist, verify that it has a REG_DWORD value of `0`. If the value is instead set to `1`, it must be changed to `0`. The value can be changed by running the following command from an elevated command prompt:

  Windows Command Prompt

  ```
  reg.exe add HKLM\SOFTWARE\Policies\Microsoft\Windows\WindowsUpdate /v
  DoNotConnectToWindowsUpdateInternetLocations /t REG_DWORD /d 1 /f
  ```

  > **Note**
  >
  > Make sure to first check the group policy of **Do not connect to any Windows Update Internet locations**. If the policy is **Enabled**, then this registry key will eventually be reset back to `1` even after it's manually set to `0` via `reg.exe`. Setting the policy of **Do not connect to any Windows Update Internet locations** to **Disabled** or **Not Configured** will make sure the registry value remains as `0`.

Delay in the activation of Enterprise license of Windows

There might be a delay in the activation of the Enterprise license in Windows. This delay is by design. Windows uses a built-in cache when determining upgrade eligibility. This behavior includes processing responses that indicate that the device isn't eligible for an upgrade. It can take up to four days after a qualifying purchase before the upgrade eligibility is enabled and the cache expires.

Virtual Desktop Access (VDA)

Subscriptions to Windows Enterprise are also available for virtualized clients. Enterprise E3 and E5 are available for Virtual Desktop Access (VDA) in Azure or in another qualified multitenant host.

Virtual machines (VMs) must be configured to enable Windows Enterprise subscriptions for VDA. Active Directory-joined and Microsoft Entra joined clients are supported. For more information, see Enable VDA for Enterprise subscription activation.

Related content

- Windows subscription activation.
- MDM enrollment of Windows devices.

Last updated on 05/14/2025

Volume activation for Windows

Applies to: ✅ Windows 11, ✅ Windows 10, ✅ Windows Server 2022, ✅ Windows Server 2019, ✅ Windows Server 2016

> **Tip**
>
> Looking for volume licensing information?
>
> - Download the Microsoft Volume Licensing Reference Guide.
>
> Looking for information on retail activation?
>
> - Activate Windows.
> - Product activation for Windows.

This guide is designed to help organizations that are planning to use volume activation to deploy and activate Windows.

Volume activation is the process that Microsoft volume licensing customers use to automate and manage the activation of Windows operating systems, Microsoft Office, and other Microsoft products across large organizations. Volume licensing is available to customers who purchase software under various volume programs (such as Open and Select) and to participants in programs such as the Microsoft Partner Program and Visual Studio Codespace.

Volume activation is a solution that automates and manages product activation on computers running Windows that are licensed under a volume licensing program. Volume activation is also used with other software from Microsoft (most notably the Office suites) that are sold under volume licensing agreements and that support volume activation.

This guide provides information and step-by-step guidance to help choose a volume activation method that suits an environment, and then to configure that solution successfully. This guide describes the volume activation features and the tools to manage volume activation.

The following items are needed to successfully plan and implement a volume activation strategy:

- Learn about and understand product activation.
- Review and evaluate the available activation types or models.
- Consider the connectivity of the clients to be activated.
- Choose the method or methods to be used with each type of client.
- Determine the types and number of product keys needed.

- Determine the monitoring and reporting needs in the organization.
- Install and configure the tools required to support the methods selected.

Keep in mind that the method of activation doesn't change an organization's responsibility to the licensing requirements. Ensure that all software used in an organization is properly licensed and activated in accordance with the terms of the licensing agreements in place.

Related content

- Plan for volume activation.
- Activate using Key Management Service.
- Activate using Active Directory-based activation.
- Activate clients running Windows.
- Monitor activation.
- Use the Volume Activation Management Tool.
- Appendix: Information sent to Microsoft during activation.

Last updated on 05/14/2025

Plan for volume activation

Applies to: ✅ Windows 11, ✅ Windows 10, ✅ Windows Server 2022, ✅ Windows Server 2019, ✅ Windows Server 2016

> **Tip**
>
> Looking for information on retail activation?
>
> - Activate Windows.
> - Product activation for Windows.

Product activation is the process of validating software with the manufacturer after it's installed on a specific computer. Activation confirms that the product is genuine and not a fraudulent copy. Activation also confirms that the product key or serial number is valid and isn't compromised or revoked. Activation also establishes a link or relationship between the product key and the particular installation.

During the activation process, information about the specific installation is examined. For online activations, this information is sent to a server at Microsoft. This information might include the software version, the product key, the IP address of the computer, and information about the device. The activation methods that Microsoft uses are designed to help protect user privacy, and they can't be used to track back to the computer or user. The gathered data confirms that the software is a legally licensed copy, and this data is used for statistical analysis. Microsoft doesn't use this information to identify or contact the user or the organization.

> **Note**
>
> The IP address is used only to verify the location of the request, because some editions of Windows (such as "Starter" editions) can only be activated within certain geographical target markets.

Distribution channels and activation

In general, Microsoft software is obtained through three main channels: retail, original equipment manufacturer (OEM), and volume licensing agreements. Different activations methods are available through each channel. Because organizations are free to obtain software through multiple channels (for example, buying some at retail and others through a volume licensing program) most organizations choose to use a combination of activation methods.

Retail activations

For retail activation, each purchased copy comes with one unique product key, often referred to as a retail key. The user enters this key during product installation. The computer uses this retail key to complete the activation after the installation is complete. Most activations are performed online, but telephone activation is also available.

Other distribution scenarios also exist. Product key cards are available to activate products that are preinstalled or downloaded. Programs such as Windows Anytime Upgrade and Get Genuine allow users to acquire legal keys separately from the software. These electronically distributed keys might come with media that contains software, they can come as a software shipment, or they might be provided on a printed card or electronic copy. Products are activated the same way with any of these retail keys.

Original equipment manufacturer

Most original equipment manufacturers (OEMs) sell systems that include a standard build of the Windows operating system. The hardware vendor activates Windows by associating the operating system with the firmware/BIOS of the computer. This activation occurs before the computer is sent to the customer, and no additional actions are required.

OEM activation is valid as long as the customer uses the OEM-provided image on the system. OEM activation is available only for computers that are purchased through OEM channels and have the Windows operating system preinstalled.

Volume licensing

Volume licensing offers customized programs that are tailored to the size and purchasing preference of the organization. To become a volume licensing customer, the organization must set up a volume licensing agreement with Microsoft. There's a common misunderstanding about acquiring licenses for a new computer through volume licensing. There are two legal ways to acquire a full Windows client license for a new computer:

- Have the license preinstalled through the OEM.
- Purchase a fully packaged retail product.

The licenses that are provided through volume licensing programs such as Open License, Select License, and Enterprise Agreements cover upgrades to Windows client operating systems only. Before the upgrade rights obtained through volume licensing can be exercised, an existing retail or OEM operating system license is needed for each computer running currently supported versions of Windows.

Volume licensing is also available through certain subscription or membership programs, such as the Microsoft Partner Network and Visual Studio Codespace. These volume licenses might contain specific restrictions or other changes to the general terms applicable to volume licensing.

> **Note**
>
> Some editions of the operating system, such as Windows Enterprise, and some editions of application software are available only through volume licensing agreements or subscriptions.

Activation models

For a user or IT department, there are no significant choices about how to activate products that are acquired through retail or OEM channels. The OEM performs the activation at the factory, and the user or the IT department don't need to take any activation steps.

With a retail product, the Volume Activation Management Tool (VAMT), which is discussed later in this guide, helps track and manage keys. For each retail activation, the following options can be chosen:

- Online activation.
- Telephone activation.
- VAMT proxy activation.

Telephone activation is primarily used in situations where a computer is isolated from all networks. VAMT proxy activation with retail keys is sometimes used when an IT department wants to centralize retail activations. VAMT can also be used when a computer with a retail version of the operating system is isolated from the internet but connected to the LAN. For volume-licensed products, however, the best method or combination of methods must be determined to use in the environment. For currently supported versions of Windows Pro and Enterprise, one of the following three models can be chosen:

- Multiple Activation Keys (MAK).
- KMS.
- Active Directory-based activation.

> **Note**
>
> Token-based activation for Windows Enterprise (including LTSC) and Windows Server is available for specific situations when approved customers rely on a public key

infrastructure in an isolated and high-security environment. For more information, contact the Microsoft Account Team or service representative.

Multiple activation key

A Multiple Activation Key (MAK) is commonly used in small or mid-sized organizations that have a volume licensing agreement, but don't meet the requirements to operate a KMS. MAK can also be used if a simpler approach is preferred. A MAK also allows permanent activation of:

- Computers that are isolated from the KMS.
- Computers that are part of an isolated network that doesn't have enough computers to use the KMS.

To use a MAK, the computers to be activated must have a MAK installed. The MAK is used for one-time activation with the Microsoft online hosted activation services, by telephone, or by using VAMT proxy activation.

In the simplest terms, a MAK acts like a retail key, except that a MAK is valid for activating multiple computers. Each MAK can be used a specific number of times. The VAMT can help with tracking the number of performed activations with each key and how many activations remain.

Organizations can download MAK and KMS keys from the Volume Licensing Service Center website. Each MAK has a preset number of activations, which are based on a percentage of the count of licenses the organization purchases. However, the number of activations that are available can be increased with the MAK by calling Microsoft.

Key Management Service

With the Key Management Service (KMS), IT pros can complete activations on their local network, eliminating the need for individual computers to connect to Microsoft for product activation. The KMS is a lightweight service that doesn't require a dedicated system and can easily be cohosted on a system that provides other services.

Volume editions of currently supported versions of Windows and Windows Server automatically connect to a system that hosts the KMS to request activation. No action is required from the user.

The KMS requires a minimum number of computers, either physical computers or virtual machines, in a network environment. The organization must have at least five computers to activate currently supported versions of Windows Server and at least 25 computers to activate

client computers running currently supported versions of Windows client. These minimums are referred to as *activation thresholds*.

Planning to use the KMS includes selecting the best location for the KMS host and how many KMS hosts to have. One KMS host can handle a large number of activations, but organizations often deploy two KMS hosts to ensure availability. The KMS can be hosted on a client computer or on a server. Setting up KMS is discussed later in this guide.

Active Directory-based activation

Active Directory-based activation is similar to activation by using the KMS, but the activated computer doesn't need to maintain periodic connectivity with the KMS host. Instead, a domain-joined computer running currently supported versions of Windows or Windows Server queries ADDS for a volume activation object that is stored in the domain. The operating system checks the digital signatures that are contained in the activation object, and then activates the device.

Active Directory-based activation allows enterprises to activate computers through a connection to their domain. Many companies have computers at remote or branch locations, where it's impractical to connect to a KMS, or wouldn't reach the KMS activation threshold. Rather than use MAK, Active Directory-based activation provides a way to activate computers running currently supported versions of Windows and Windows Server as long as the computers can contact the company's domain. Active Directory-based activation offers the advantage of extending volume activation services everywhere where there's already a domain presence.

Network and connectivity

A modern business network has many nuances and interconnections. This section examines evaluating the organization's network and the connections that are available to determine how volume activations occur.

Core network

The organization's core network is that part of the network that enjoys stable, high-speed, reliable connectivity to infrastructure servers. In many cases, the core network is also connected to the internet. However, internet connectivity isn't a requirement to use the KMS or Active Directory-based activation after the KMS server or ADDS is configured and active. The organization's core network likely consists of many network segments. In many organizations, the core network makes up most of the business network.

In the core network, a centralized KMS solution is recommended. Active Directory-based activation can also be used, but in many organizations, KMS might still be required to computers that aren't joined to the domain. Some administrators prefer to run both solutions to have the most flexibility, while others prefer to choose only a KMS-based solution for simplicity. Active Directory-based activation as the only solution is workable if all of the clients in the organization are running currently supported versions of Windows.

A typical core network that includes a KMS host is shown in Figure 1.

Figure 1. Typical core network

Isolated networks

In a large network, some segments might be isolated, either for security reasons or because of geography or connectivity issues.

Isolated for security

A network segment isolated from the core network by a firewall or disconnected from other networks is sometimes called a *high-security zone*. The best solution for activating computers in an isolated network depends on the security policies in place in the organization.

If the isolated network can:

- Access the core network by using outbound requests on TCP port 1688
- Allowed to receive remote procedure calls (RPCs)

activation can be performed by using the KMS in the core network, avoiding the need to reach additional activation thresholds.

If the isolated network participates fully in the corporate forest, and it can make typical connections to domain controllers, such as:

- Using Lightweight Directory Access Protocol (LDAP) for queries
- Using Domain Name Service (DNS) for name resolution

then this scenario is a good opportunity to use Active Directory-based activation for currently supported versions of Windows and Windows Server.

If the isolated network can't communicate with the core network's KMS server, and it can't use Active Directory-based activation, a KMS host can be set up in the isolated network. This configuration is shown in Figure 2. However, if the isolated network contains only a few computers, it won't reach the KMS activation threshold. In that case, MAKs can be used for activation.

If the network is fully isolated, MAK-independent activation would be the recommended choice, perhaps using the telephone option, but VAMT proxy activation might also be possible. MAKs can also be used to activate new computers during setup, before they're placed in the isolated network.

Figure 2. New KMS host in an isolated network

Branch offices and distant networks

From mining operations to ships at sea, organizations often have a few computers that aren't easily connected to the core network or the internet. Some organizations have network segments at branch offices that are large and well-connected internally, but have a slow or unreliable WAN link to the rest of the organization. There are several options in these situations:

- **Active Directory-based activation**. In any site where the client computers are running currently supported versions of Windows, Active Directory-based activation is supported, and it can be activated by joining the domain.

- **Local KMS**. If a site has 25 or more client computers, it can activate against a local KMS server.
- **Remote (core) KMS**. If the remote site has connectivity to an existing KMS, perhaps through a virtual private network (VPN) to the core network, that KMS can be used. Using the existing KMS means that the activation threshold only needs to be met on that server.
- **MAK activation**. If the site has only a few computers and no connectivity to an existing KMS host, MAK activation is the best option.

Disconnected computers

Some users might be in remote locations or might travel to many locations. This scenario is common for roaming clients, such as the computers that are used by salespeople or other users who are offsite but not at branch locations. This scenario can also apply to remote branch office locations that have no connection to the core network. This branch office can be considered an "isolated network," where the number of computers is one. Disconnected computers can use Active Directory-based activation, the KMS, or MAK depending on how often the computers connect to the core network.

Active Directory-based activation can be used on computers when they meet the following conditions:

- The computer is joined to the domain.
- The computer is running a currently supported version of Windows or Windows Server.
- The computer connects to the domain at least once every 180 days, either directly or through a VPN.

Otherwise for computers that rarely or never connect to the network, MAK independent activation should be used either via the telephone or the internet.

Test and development labs

Lab environments often have large numbers of virtual machines, and physical computers and virtual machines in labs are reconfigured frequently. Therefore, first determine whether the computers in test and development labs require activation. Currently supported editions of Windows that include volume licensing operate normally, even if they can't activate immediately.

If the test or development copies of the operating system are within the license agreement, the lab computers might not need to be activated if they're rebuilt frequently. If the lab computers need to be activated, treat the lab as an isolated network, and use the methods described earlier in this guide. In labs that have a high turnover of computers and a few KMS clients, the

KMS activation count must be monitored. The time that the KMS caches the activation requests might need to be adjusted. The default is 30 days.

Mapping the network to activation methods

By evaluating network connectivity and the numbers of computers at each site, the information needed to determine which activation methods work best can be determined. This information can be filled in Table 1 to help make this determination.

Table 1. Criteria for activation methods

Expand table

Criterion	Activation method
Number of domain-joined computers that will connect to a domain controller at least every 180 days. Computers can be mobile, semi-isolated, or located in a branch office or the core network.	Active Directory-based activation
Number of computers in the core network that will connect at least every 180 days, either directly or through VPN. The core network must meet the KMS activation threshold.	KMS (central)
Number of computers that don't connect to the network at least once every 180 days, or if no network meets the activation threshold.	MAK
Number of computers in semi-isolated networks that have connectivity to the KMS in the core network.	KMS (central)
Number of computers in isolated networks where the KMS activation threshold is met.	KMS (local)
Number of computers in isolated networks where the KMS activation threshold isn't met.	MAK
Number of computers in test and development labs that won't be activated.	None
Number of computers that don't have a retail volume license.	Retail (online or phone)
Number of computers that don't have an OEM volume license.	OEM (at factory)
Total number of computer activations. This total should match the total number of licensed computers in the organization.	

Choosing and acquiring keys

When it's know which keys are needed, the keys must be obtained. Generally speaking, volume licensing keys are collected in two ways:

- Go to the **Product Keys** section of the Volume Licensing Service Center for the following agreements: Open, Open Value, Select, Enterprise, and Services Provider License.
- Contact the Microsoft activation center.

KMS host keys

A KMS host needs a key that activates, or authenticates, the KMS host with Microsoft. This key is referred to as the *KMS host key*, but it's formally known as a *Microsoft Customer Specific Volume License Key* (CSVLK). Some documentation and internet references use the term KMS key, but CSVLK is the proper name for current documentation and management tools.

A KMS host running a currently supported version of Windows Server can activate both Windows Server and Windows client operating systems. A KMS host key is also needed to create the activation objects in ADDS, as described later in this guide. A KMS host key is needed for any KMS that is set up. Additionally, it needs to be determined if Active Directory-based activation will be used.

Generic volume licensing keys

If computers are activated with KMS or Active Directory-based activation when using custom installation media or an image to install Windows, install a generic volume license key (GVLK) when creating the custom installation media or image. The GVLK should match the edition of Windows being installed.

Installation media from Microsoft for Enterprise editions of the Windows operating system might already contain the GVLK. One GVLK is available for each type of installation. The GVLK doesn't activate the software against Microsoft activation servers, but rather against a KMS or Active Directory-based activation object. In other words, the GVLK doesn't work unless a valid KMS host key can be found. GVLKs are the only product keys that don't need to be kept confidential.

Typically, a GVLK doesn't need to be manually entered unless a computer is:

- Activated with a MAK or a retail key.
- Being converted to a KMS activation or to Active Directory-based activation.

If the GVLK for a particular client edition needs to be located, see Key Management Services (KMS) client activation and product keys.

Multiple activation keys

MAK keys with the appropriate number of activations available are also needed. The number of times a MAK has been used can be seen on the Volume Licensing Service Center website or in the VAMT.

Selecting a KMS host

The KMS doesn't require a dedicated server. It can be cohosted with other services, such as ADDS domain controllers and read-only domain controllers.

KMS hosts can run on physical computers or virtual machines that are running any supported Windows operating system. A KMS host that is running currently supported versions of Windows Server can activate any Windows client or server operating system that supports volume activation. A KMS host that is running a currently supported version of Windows client can only activate computers running a currently supported version of Windows client.

A single KMS host can support unlimited numbers of KMS clients, but Microsoft recommends deploying a minimum of two KMS hosts for failover purposes. However, as more clients are activated through Active Directory-based activation, the KMS and the redundancy of the KMS might not be needed. Most organizations can use as few as two KMS hosts for their entire infrastructure.

The flow of KMS activation is shown in Figure 3, and it follows this sequence:

1. An administrator uses the VAMT console to configure a KMS host and install a KMS host key.
2. Microsoft validates the KMS host key, and the KMS host starts to listen for requests.
3. The KMS host updates resource records in DNS to allow clients to locate the KMS host. Manually adding DNS records is required if the environment doesn't support DNS dynamic update protocol.
4. A client configured with a GVLK uses DNS to locate the KMS host.
5. The client sends one packet to the KMS host.
6. The KMS host records information about the requesting client (by using a client ID). Client IDs are used to maintain the count of clients and detect when the same computer is requesting activation again. The client ID is only used to determine whether the activation thresholds are met. The IDs aren't stored permanently or transmitted to Microsoft. If the KMS is restarted, the client ID collection starts again.

7. If the KMS host has a KMS host key that matches the products in the GVLK, the KMS host sends a single packet back to the client. This packet contains a count of the number of computers that requested activation from this KMS host.

8. If the count exceeds the activation threshold for the product that is being activated, the client is activated. If the activation threshold isn't met, the client tries again.

Figure 3. KMS activation flow

Related content

- Volume activation for Windows.

Last updated on 05/14/2025

Activate using Key Management Service

Applies to: ✅ Windows 11, ✅ Windows 10, ✅ Windows Server 2022, ✅ Windows Server 2019, ✅ Windows Server 2016

> **Tip**
>
> For information on retail activation, see the following articles:
>
> - Activate Windows
> - Product activation for Windows

Volume activation can be performed via Key Management Service (KMS). KMS can be hosted either on a client version of Windows or on Windows Server.

Key Management Service in a client version of Windows

Installing a KMS host key on a computer running a client version of Windows allows the following scenarios against this KMS host:

- Activation of other computers running the same client version of Windows.
- Activation of other computers running earlier client versions of Windows.

Clients locate the KMS server by using resource records in DNS, so some configuration of DNS is required. This scenario can be beneficial if the organization uses volume activation for clients and MAK-based activation for a smaller number of servers.

To enable KMS functionality, a KMS key is installed on a KMS host. The host is then activated over the Internet or by phone using Microsoft activation services.

Configure KMS in a client version of Windows

KMS can be activated on client versions of Windows by using the `slmgr.vbs`. To activate KMS on a client version of Windows, follow these steps:

1. Open an elevated Command Prompt window.

2. In the elevated Command Prompt window, run the following command to install the KMS key:

Windows Command Prompt

```
cscript.exe slmgr.vbs /ipk <KMS_Key>
```

3. Once the KMS key is installed, it needs to be activated using one of the following methods:

   - To activate online, in the elevated Command Prompt window, run the following command:

Windows Command Prompt

```
cscript.exe slmgr.vbs /ato
```

   - To activate by telephone, follow these steps:

     a. In the elevated Command Prompt window, run the following command:

Windows Command Prompt

```
cscript.exe slmgr.vbs /dti
```

     This command should display the installation ID.

     b. Call the Microsoft Volume License Key assisted support telephone numbers. Follow the voice prompts and when prompted, enter the installation ID obtained in the previous step.

     c. Continue following the voice prompts. When prompted, write down the 48-digit confirmation ID for OS activation given by the prompts.

     d. In the elevated Command Prompt window, run the following command:

Windows Command Prompt

```
cscript.exe slmgr.vbs /atp <confirmation_ID_from_previous_step>
```

Key Management Service in Windows Server

Installing a KMS host key on a computer running Windows Server allows activation of computers running the same or earlier versions of Windows Server. Additionally, it also allows activation of client versions of Windows.

> **Important**

A client KMS key can't be installed into the KMS in Windows Server.

Configure KMS in Windows Server

1. Sign in to a Windows Server server with an account that has local administrative credentials.

2. Open **Server Manager**.

3. Under the **Manage** menu in **Server Manager**, select **Add Roles and Features**. The **Add Roles and Features Wizard** window opens.

4. In the **Add Roles and Features Wizard**:

   a. In the **Before you begin** page, select the **Next >** button.

   b. In the **Select installation type/Installation Type** page, select **Role-based or feature-based installation**, and then select the **Next >** button.

   c. In the **Select destination server/Server Selection** page, make sure **Select a server from the server pool** is selected. Under **Server Pool**, select the server on which to install KMS, and then select the **Next >** button.

   d. In the **Select server roles/Server Roles** page, under **Roles**, select **Volume Activation Services**, and then select the **Next >** button.

   e. In the **Add features that are required for Volume Activation Services?** window that appears, select the **Add Features** button, and then select the **Next >** button.

   f. In the **Select features/Features** page, select the **Next >** button.

   g. In the **Volume Activation Services** page, select the **Next >** button.

   h. In the **Confirm installation selections/Confirmation** page, select the **Install** button.

   i. Installation can take a few minutes to complete. Once the role installation completes, select the **Close** button.

5. Go to the **Start Menu** > **Windows Administrative Tools** and select **Volume Activation Tools**. The **Volume Activation Tools** window appears.

6. In the **Volume Activation Tools** window:

   a. In the **Introduction to Volume Activation Tools/Introduction** page, select the **Next >** button.

b. In the **Select Volume Activation Method/Activation Type** page, select the **Key Management Service (KMS)** option, and specify the computer that acts as the KMS host. This computer can be the server on which the KMS role was installed, or another server/client computer. After the server/computer is specified, select the **Next >** button.

c. In the **Manage KMS Host/Product Key Management** page, enter in the KMS host key in the text box under **Install your KMS host key**, and then select the **Commit** button.

d. If asked to confirm replacement of an existing key, select **Yes**.

e. After the product key is installed, in the **Product Key Installation Succeeded/Product Key Management** page, make sure **Activate Product** is selected, and then select **Next >** button to begin the activation process.

f. In the **Activate Product/Product Key Management** page, make sure the current product is shown under the **Select product** menu, and then select the desired activation method. The available methods are:

- **Active online** - If selecting this option, select the **Commit** button to finish activating the product online.
- **Active by phone** - If selecting this option:

  i. Select the desired location from the **Select your location** drop-down menu, and then select the **Next >** button.

  ii. In the **Activate by Phone/Product Key Management** page, follow the instructions to activate the product by phone.

  iii. Once finished, select the **Commit** button.

g. In the **Activation Succeeded/Product Key Management** page, review the configuration options:

- If the configuration options are as expected, select the **Close** button.
- If configuration changes are desired:

  i. Select the **Next >** button.

  ii. In the **Configure Key Management Service Options/Product Key Management** page, make the desired configuration changes, and then select the **Commit** button.

  iii. In the **Configuration Succeeded/Configuration** page, select the **Close** button.

Once the KMS host is configured, it begins to listen for activation requests. However, it doesn't activate clients successfully until the activation threshold is met.

Verifying the configuration of Key Management Service

KMS volume activation can be verified from the KMS host server or from the client computer. KMS volume activation requires a minimum threshold of 25 computers before activation requests are processed. The verification process described here increments the activation count each time a client computer contacts the KMS host. If the activation threshold hasn't been reached, the verification generates an error message instead of a confirmation message.

> **Note**
>
> If Active Directory-based activation was configured before configuring KMS activation, a client computer must be used that doesn't first try to activate itself by using Active Directory-based activation. For example, a client computer that is a workgroup computer that isn't joined to a domain.

To verify that KMS volume activation works, complete the following steps:

1. On the KMS host, open the event log and confirm that DNS publishing is successful.
2. On a client computer, open an elevated Command Prompt window and run the command:

   Windows Command Prompt
   ```
   cscript.exe slmgr.vbs /ato
   ```

   The `/ato` command causes the operating system to attempt activation by using whichever key is installed in the operating system. The response should show the license state and detailed Windows version information.
3. On a client computer or the KMS host, open an elevated Command Prompt window and run the command

   Windows Command Prompt
   ```
   cscript.exe slmgr.vbs /dlv
   ```

The `/dlv` command displays the detailed licensing information. The response should return an error that states that the KMS activation count is too low. This test confirms that KMS is functioning correctly, even though the client isn't activated.

For more information about the use and syntax of the script `slmgr.vbs`, see Slmgr.vbs Options.

> **Important**
>
> Clients require RPC over TCP/IP connectivity to the KMS host to successfully activate. For more information, see Key Management Services (KMS) activation planning: Network requirements and Remote Procedure Call (RPC) errors troubleshooting guidance.

Related content

- Key Management Services (KMS) activation planning.

Last updated on 05/14/2025

Activate using Active Directory-based activation

Applies to: ✅ Windows 11, ✅ Windows 10, ✅ Windows Server 2022, ✅ Windows Server 2019, ✅ Windows Server 2016, ✅ Microsoft Office

> **Tip**
>
> Looking for information on retail activation?
>
> - Activate Windows.
> - Product activation for Windows.

Active Directory-based activation is implemented as a role service that relies on Active Directory Domain Services (ADDS) to store activation objects. Active Directory-based activation requires updating the forest schema with `adprep.exe` on a supported server OS. After the schema is updated, older domain controllers can still activate clients.

Any domain-joined computers running a supported OS with a Generic Volume License Key (GVLK) is activated automatically and transparently. Domain-joined computers stay activated as long as they remain members of the domain and maintain periodic contact with a domain controller. Activation takes place after the Licensing service starts. When this service starts, the computer contacts ADDS automatically, receives the activation object, and is activated without user intervention.

To allow computers with GVLKs to activate themselves, use the Volume Activation Tools console, or the Volume Activation Management Tool (VAMT) in earlier versions of Windows Server to create an object in the ADDS forest. The activation object is created by submitting a Key Management Service (KMS) host key to Microsoft, as shown in Figure 10.

The process proceeds as follows:

1. Do *one* of the following tasks:

    - Install the Volume Activation Services server role on a domain controller, then add a KMS host key by using the Volume Activation Tools Wizard.

    - Extend the domain schema level to Windows Server 2012 R2 or later, then add a KMS host key by using the VAMT.

2. Microsoft verifies the KMS host key, and an activation object is created.

3. Client computers are activated by receiving the activation object from a domain controller during startup.

Figure 10. The Active Directory-based activation flow

For environments where all computers are domain joined and running a supported OS version, Active Directory-based activation is the best option for activating client computers and servers. Active Directory-based activation might allow removal of any KMS hosts from the environment. If an environment contains one of the following items:

- Earlier versions of volume licensed operating systems and applications
- Workgroup computers outside the domain

a KMS host is still needed to maintain activation status.

Clients that are activated with Active Directory-based activation maintain their activated state for up to 180 days since the last contact with the domain. They periodically attempt to reactivate before then and at the end of the 180 day period. By default, this reactivation event occurs every seven days.

When a reactivation event occurs, the client queries ADDS for the activation object. Client computers examine the activation object and compare it to the local edition as defined by the GVLK. If the object and the GVLK match, then reactivation occurs. If the ADDS object can't be retrieved, client computers use KMS activation. If the computer is removed from the domain and the computer or the Software Protection service is restarted, Windows changes the status to **Not Activated** and the computer tries to activate with KMS.

Step-by-step configuration: Active Directory-based activation

> **Note**
>
> The administrator following these steps must be a member of the local **Administrators** group on all computers mentioned in these steps. Additionally, they also need to be a member of the **Enterprise Administrators** group, because setting up Active Directory-based activation changes forest-wide settings.

To configure Active Directory-based activation on a supported version of Windows Server, complete the following steps:

1. Use an account with **Domain Administrator** and **Enterprise Administrator** credentials to sign in to a domain controller.

2. Launch **Server Manager**.

3. Add the **Volume Activation Services** role, as shown in Figure 11.

Figure 11. Adding the Volume Activation Services role

4. Select the **Volume Activation Tools**, as shown in Figure 12.

Figure 12. Launching the Volume Activation Tools

5. Select the **Active Directory-Based Activation** option, as shown in Figure 13.

Figure 13. Selecting Active Directory-Based Activation

6. Enter the organization's KMS host key and optionally specify a display name, as shown in Figure 14.

Figure 14. Entering the organization's KMS host key

7. Activate the organization's KMS host key by phone or online, as shown in Figure 15.

Figure 15. Choosing how to activate the product

> **Note**
>
> To activate a KMS Host Key/Customer Specific Volume License Key (CSVLK) for Microsoft Office, the version-specific Office Volume License Pack needs to be installed on the server where the Volume Activation Server Role is installed.
>
> - Office 2016 VL pack.
> - Office 2019 VL pack.
> - Office LTSC 2021 VL pack.
>
> For more information, see Activate volume licensed versions of Office by using Active Directory.

8. After activating the key, select **Commit**, and then select **Close**.

Verifying the configuration of Active Directory-based activation

To verify the Active Directory-based activation configuration, complete the following steps:

1. After configuring Active Directory-based activation, start a computer running an edition of Windows configured by volume licensing.
2. If the computer was previously configured with a MAK key, replace the MAK key with the GVLK. Run the `slmgr.vbs /ipk` command and specifying the GVLK as the new product key.
3. If the computer isn't joined to the organization's domain, join it to the domain.
4. Sign in to the computer.
5. Open Windows Explorer, right-click **Computer**, and then select **Properties**.
6. Scroll down to the **Windows activation** section, and verify that this client is activated.

> **Note**
>
> If using both KMS and Active Directory-based activation, it might be difficult to determine is a client was activated with KMS or by Active Directory-based activation. During the test, consider disabling KMS, or ensure to use a client computer not

already activated by KMS. The `slmgr.vbs /dlv` command also indicates if KMS was used.

To manage individual activations or apply multiple (mass) activations, use the **VAMT**.

Related content

- Volume Activation for Windows.

Last updated on 05/14/2025

Activate clients running Windows

Applies to: ✅ Windows 11, ✅ Windows 10, ✅ Windows Server 2022, ✅ Windows Server 2019, ✅ Windows Server 2016

> **Tip**
>
> Looking for information on retail activation?
>
> - Activate Windows.
> - Product activation for Windows.

After Key Management Service (KMS) or Active Directory-based activation is configured in a network, activating a client running Windows is easy. If the computer is configured with a Generic Volume License Key (GVLK), IT or the user don't need to take any action. It just works.

Enterprise edition images and installation media should already be configured with the GVLK. When the client computer starts, the Licensing service examines the current licensing condition of the computer.

If activation or reactivation is required, the following sequence occurs:

1. If the computer is a member of a domain, it asks a domain controller for a volume activation object. If Active Directory-based activation is configured, the domain controller returns the object. If the object meets the following requirements:

    - Matches the edition of the software that is installed
    - Has a matching GVLK

then the computer is activated (or reactivated). The computer doesn't need to activate again for 180 days although the operating system attempts reactivation at shorter, regular intervals.

1. If the computer isn't a member of a domain or if the volume activation object isn't available, the computer issues a DNS query to attempt to locate a KMS server. If a KMS server can be contacted, activation occurs if the KMS has a key that matches the computer's GVLK.

2. The computer tries to activate against Microsoft servers if it's configured with a MAK.

If the client isn't able to activate itself successfully, it periodically tries again. The frequency of the retry attempts depends on the current licensing state and whether the client computer successfully activated in the past. For example, if the client computer previously used Active

Directory-based activation to activate, it periodically tries to contact the domain controller at each restart.

How Key Management Service works

KMS uses a client-server topology. KMS client computers can locate KMS host computers by using DNS or a static configuration. KMS clients contact the KMS host by using RPCs carried over TCP/IP.

Key Management Service activation thresholds

Physical computers and virtual machines can activate by contacting a KMS host. To qualify for KMS activation, there must be a minimum number of qualifying computers. This minimum is called the activation threshold. KMS clients will be activated only after this threshold is met. Each KMS host counts the number of computers that requested activation until the threshold is met.

A KMS host responds to each valid activation request from a KMS client with the count of how many computers have already contacted the KMS host for activation. Client computers that receive a count below the activation threshold aren't activated. For example, if the first two computers that contact the KMS host are running a currently supported version of Windows client, the first receives an activation count of 1, and the second receives an activation count of 2. If the next computer is a virtual machine running a currently supported version of Windows client, it receives an activation count of 3, and so on. None of these computers are activated because an activation count of 25 or more must be reached.

When KMS clients are waiting for the KMS to reach the activation threshold, they connect to the KMS host every two hours to get the current activation count. They're activated once the threshold is met.

In our example, if the next computer that contacts the KMS host is running a currently supported version of Windows Server, it receives an activation count of 4 since activation counts are cumulative. If a computer running a currently supported version of Windows Server receives an activation count that is 5 or more, it's activated. If a computer running a currently supported version of Windows client receives an activation count of 25 or more, it's activated.

Activation count cache

To track the activation threshold, the KMS host keeps a record of the KMS clients that request activation. The KMS host gives each KMS client a client ID designation, and the KMS host saves each client ID in a table. By default, each activation request remains in the table for up to 30

days. When a client renews its activation, the cached client ID is removed from the table, a new record is created, and the 30 day period begins again. If a KMS client computer doesn't renew its activation within 30 days, the KMS host removes the corresponding client ID from the table and reduces the activation count by one.

However, the KMS host only caches twice the number of client IDs that are required to meet the activation threshold. Therefore, only the 50 most recent client IDs are kept in the table, and a client ID could be removed sooner than 30 days.

The type of client computer that is attempting to activate sets the total size of the cache. For example, if a KMS host receives activation requests only from servers, the cache holds only 10 client IDs, twice the required threshold of 5. However, if a client computer running Windows client contacts that KMS host, KMS increases the cache size to 50 to accommodate the higher threshold. KMS never reduces the cache size.

Key Management Service connectivity

KMS activation requires TCP/IP connectivity. By default, KMS hosts and clients use DNS to publish and find the KMS. The default settings can be used, which require little or no administrative action. However, the KMS hosts and client computers can be manually configured based on network configuration and security requirements.

Key Management Service activation renewal

KMS activations are valid for 180 days (the *activation validity interval*). To remain activated, KMS client computers must renew their activation by connecting to the KMS host at least once every 180 days. By default, KMS client computers attempt to renew their activation every seven days. If KMS activation fails, the client computer retries every two hours. After a client computer's activation is renewed, the activation validity interval begins again.

Publication of the Key Management Service

The KMS uses service (SRV) resource records in DNS to store and communicate the locations of KMS hosts. KMS hosts use the DNS dynamic update protocol, if available, to publish the KMS service (SRV) resource records. If dynamic update isn't available or the KMS host doesn't have rights to publish the resource records, one of the following actions needs to be taken:

- The DNS records must be published manually.
- Client computers must be configured to connect to specific KMS hosts.

Client discovery of the Key Management Service

By default, KMS client computers query DNS for KMS information. The first time a KMS client computer queries DNS for KMS information, it randomly chooses a KMS host from the list of service (SRV) resource records that DNS returns. The address of a DNS server that contains the service (SRV) resource records can be listed as a suffixed entry on KMS client computers. This feature allows one DNS server to advertise the service (SRV) resource records for KMS, and KMS client computers with other primary DNS servers to find it.

Priority and weight parameters can be added to the DnsDomainPublishList registry value for KMS. Establishing KMS host priority groupings and weighting within each group allows specifying which KMS host the client computers should try first and balances traffic among multiple KMS hosts. All currently supported versions of Windows and Windows Server provide these priority and weight parameters.

If the KMS host that a client computer selects doesn't respond, the KMS client computer removes that KMS host from its list of service (SRV) resource records and randomly selects another KMS host from the list. When a KMS host responds, the KMS client computer caches the name of the KMS host and uses it for subsequent activation and renewal attempts. If the cached KMS host doesn't respond on a subsequent renewal, the KMS client computer discovers a new KMS host by querying DNS for KMS service (SRV) resource records.

By default, client computers connect to the KMS host for activation by using anonymous RPCs through TCP port 1688, although the default port can be changed. After a client computer establishes a TCP session with the KMS host, the client computer sends a single request packet. The KMS host responds with the activation count. If the count meets or exceeds the activation threshold, the client computer is activated, and the session is closed. The KMS client computer uses this same process for renewal requests. 250 bytes are used for communication each way.

Domain Name System server configuration

The default KMS automatic publishing feature requires the service (SRV) resource record and support for DNS dynamic update protocol. KMS client computer default behavior and the KMS service (SRV) resource record publishing are supported on:

- A DNS server that is running Microsoft software.
- DNS server that supports service (SRV) resource records (per Internet Engineering Task Force [IETF] Request for Comments [RFC] 2782) and dynamic updates (per IETF RFC 2136).

For example, Berkeley Internet Domain Name versions 8.x and 9.x support service (SRV) resource records and dynamic update. The KMS host must be configured so that it has the credentials needed to create and update the following resource records on the DNS servers: service (SRV), IPv4 host (A), and IPv6 host (AAAA), or the records need to be created manually. The recommended solution for giving the KMS host the needed credentials is to create a

security group in AD DS, then add all KMS hosts to that group. On a DNS server that is running Microsoft software, ensure that this security group is given full control over the _VLMCS._TCP record. This requirement needs to occur in each DNS domain that contains the KMS service (SRV) resource records.

Activating the first Key Management Service host

KMS hosts on the network need to install a KMS key, and then be activated with Microsoft. Installation of a KMS key enables the KMS on the KMS host. After installing the KMS key, complete the activation of the KMS host by telephone or online. Beyond this initial activation, a KMS host doesn't communicate any information to Microsoft. KMS keys are only installed on KMS hosts, never on individual KMS client computers.

Activating subsequent Key Management Service hosts

Each KMS key can be installed on up to six KMS hosts. These hosts can be physical computers or virtual machines. After a KMS host is activated, the same host can be reactivated up to nine times with the same key. If the organization needs more than six KMS hosts, additional activations can be requested for an organization's KMS key by calling a Microsoft Volume Licensing Activation Center to request an exception.

How Multiple Activation Key works

A MAK is used for one-time activation with Microsoft's hosted activation services. Each MAK has a predetermined number of allowed activations. This number is based on volume licensing agreements, and it might not match the organization's exact license count. Each activation that uses a MAK with the Microsoft hosted activation service counts toward the activation limit.

Computers can be activated by using a MAK in two ways:

- **MAK independent activation**. Each computer independently connects and is activated with Microsoft over the Internet or by telephone. MAK independent activation is best suited to computers within an organization that don't maintain a connection to the corporate network. MAK independent activation is shown in Figure 16.

Figure 16. MAK independent activation

- **MAK proxy activation**. MAK proxy activation enables a centralized activation request on behalf of multiple computers with one connection to Microsoft. MAK proxy activation can be configured by using the VAMT. MAK proxy activation is appropriate for environments in which security concerns restrict direct access to the Internet or the corporate network. It's also suited for development and test labs that lack this connectivity. MAK proxy activation with the VAMT is shown in Figure 17.

Figure 17. MAK proxy activation with the VAMT

MAK is recommended for:

- Computers that rarely or never connect to the corporate network.
- Environments in which the number of computers that require activation doesn't meet the KMS activation threshold.

MAK can be used for individual computers or with an image that can be duplicated or installed using Microsoft deployment solutions. MAK can also be used on a computer that was originally configured to use KMS activation. Switching from KMS to a MAK is useful for moving a computer off the core network to a disconnected environment.

Multiple Activation Key (MAK) architecture and activation

MAK independent activation installs a MAK product key on a client computer. The key instructs that computer to activate itself with Microsoft servers over the Internet.

In MAK proxy activation, the VAMT:

- Installs a MAK product key on a client computer.
- Obtains the installation ID from the target computer.
- Sends the installation ID to Microsoft on behalf of the client.
- Obtains a confirmation ID.

The tool then activates the client computer by installing the confirmation ID.

Activating as a standard user

Currently supported versions of Windows don't require administrator privileges for activation. However, an administrator account is still required for other activation or license-related tasks, such as "rearm."

Related content

- Volume Activation for Windows.

Last updated on 05/14/2025

Monitor activation

Applies to: ✅ Windows 11, ✅ Windows 10, ✅ Windows Server 2022, ✅ Windows Server 2019, ✅ Windows Server 2016

> **Tip**
>
> Looking for information on retail activation?
>
> - Activate Windows.
> - Product activation for Windows.

The success of the activation process for a computer running Windows can be monitored in several ways. The most popular methods include:

- Using the Volume Licensing Service Center website to track use of MAK keys.
- Using the `Slmgr /dlv` command on a client computer or on the KMS host. For a full list of options, see Slmgr.vbs options for obtaining volume activation information.
- Using Windows Management Instrumentation (WMI) to view licensing status. WMI makes licensing status available to non-Microsoft or custom tools that can access WMI. Windows PowerShell can also be used to access WMI information.
- Most licensing actions and events are recorded in the Event log (ex: Application Log events 12288-12290).
- Microsoft System Center Operations Manager and the KMS Management Pack can provide insight and information to users of System Center Operations Manager.
- See Troubleshooting activation error codes for information about troubleshooting procedures for Multiple Activation Key (MAK) or the Key Management Service (KMS).
- The Volume Activation Management Tool (VAMT) provides a single site from which to manage and monitor volume activations. For more information, see Introduction to VAMT.

Related content

Volume activation for Windows.

Last updated on 05/14/2025

Use the Volume Activation Management Tool

Applies to: ✅ Windows 11, ✅ Windows 10, ✅ Windows Server 2022, ✅ Windows Server 2019, ✅ Windows Server 2016

> **Tip**
>
> Looking for information on retail activation?
>
> - Activate Windows.
> - Product activation for Windows.

The Volume Activation Management Tool (VAMT) provides several useful features, including the ability to perform VAMT proxy activation and to track and monitor several types of product keys.

Volume, retail, and Multiple Activation Keys (MAK) activation process for Windows, Office, and select other Microsoft products can be automated and centrally managed using VAMT. The VAMT can manage volume activation by using MAK or Key Management Service (KMS). It's a standard Microsoft Management Console snap-in.

For currently supported versions of Windows Server, VAMT can be installed directly from Server Manager by selecting the Volume Activation Services role or the Remote Server Administration Tools/Role Administration Tools/Volume Activation Tools feature.

For currently supported versions of Windows client, VAMT can be installed as part of the Windows Assessment and Deployment Kit (Windows ADK). The Windows ADK is a free download. For more information, including links to download the Windows ADK, see Download and install the Windows ADK.

Activate with VAMT

VAMT can be used to complete the activation process in products by using MAK and retail keys. Computers can be activated either individually or in groups. The VAMT enables two activation scenarios:

- **Online activation**. Online activation enables activation over the Internet any products that are installed with MAK, KMS host, or retail product keys. One or more connected computers can be activated within a network. This process requires each product communicate activation information directly to Microsoft.

- **Proxy activation**. This activation method enables volume activation for products that are installed on client computers that don't have Internet access. The VAMT host computer distributes a MAK, KMS host key, or retail product key to one or more client products and collects the installation ID from each client product. The VAMT host sends the installation IDs to Microsoft on behalf of the client products and obtains the corresponding confirmation IDs. The VAMT host then installs the confirmation IDs on the client products to complete their activation.

  When this method is used, only the VAMT host computer requires Internet access. Proxy activation by using the VAMT is beneficial for isolated network segments and for cases where the organization has a mix of retail, MAK, and KMS-based activations.

Track products and computers

The VAMT provides an overview of the activation and licensing status of computers across an organization's network, as shown in Figure 18. Several prebuilt reports are also available to help proactively manage licensing.

Figure 18. The VAMT showing the licensing status of multiple computers

Track key usage

The VAMT makes it easier to track the various keys that are issued to an organization. Each key can be entered into VAMT, and then the VAMT can use those keys for online or proxy activation of clients. The tool can also describe what type of key it's and to which product group it belongs. The VAMT is the most convenient way to quickly determine how many activations remain on a MAK. Figure 19 shows an example of key types and usage.

Figure 19. The VAMT showing key types and usage

Other features

The VAMT stores information in a Microsoft SQL Server database for performance and flexibility, and it provides a single graphical user interface for managing activations and performing other activation-related tasks, such as:

- **Adding and removing computers**. VAMT can be used to discover computers in the local environment. The VAMT can discover computers by querying AD DS, workgroups, or individual computer names or IP addresses, or through a general LDAP query.
- **Discovering products**. VAMT can be used to discover Windows, Windows Server, Office, and select other products that are installed on the client computers.
- **Managing activation data**. The VAMT stores activation data in a SQL Server database. The tool can export this data in XML format to other VAMT hosts or to an archive.

For more information, see:

- Volume Activation Management Tool (VAMT) Overview.

Related content

- Volume Activation for Windows.

Last updated on 05/14/2025

Appendix: Information sent to Microsoft during activation

Article • 07/25/2024 •

Applies to: ✅ Windows 11, ✅ Windows 10, ✅ Windows Server 2022, ✅ Windows Server 2019, ✅ Windows Server 2016

> **Tip**
>
> Looking for information on retail activation?
>
> - Activate Windows.
> - Product activation for Windows.

When a computer running a currently supported version of Windows is activated, the following information is sent to Microsoft:

- The Microsoft product code (a five-digit code that identifies the Windows product being activated).
- A channel ID or site code that identifies how the Windows product was originally obtained. For example, a channel ID or site code identifies whether the product was:
  - Originally purchased from a retail store.
  - Obtained as an evaluation copy.
  - Obtained through a volume licensing program.
  - Preinstalled by a computer manufacturer.
- The date of installation and whether the installation was successful.
- Information that helps confirm that the Windows product key isn't altered.
- Computer make and model.
- Version information for the operating system and software.
- Region and language settings.
- A unique number called a *globally unique identifier* (GUID), which is assigned to the computer.
- Product key (hashed) and product ID.

- BIOS name, revision number, and revision date.
- Volume serial number (hashed) of the hard disk drive.
- The result of the activation check.

  This result includes error codes and the following information about any activation exploits and related malicious or unauthorized software that was found or disabled:

  - The identifier of the activation exploit.
  - The current state of the activation exploit, such as cleaned or quarantined.
  - Computer manufacturer's identification.
  - The file name and hash of the activation exploit in addition to a hash of related software components that might indicate the presence of an activation exploit.
- The name and a hash of the contents of the computer's startup instructions file.
- If the Windows license is on a subscription basis, information about how the subscription works.

Standard computer information is also sent, but the computer's IP address is only kept temporarily.

Use of information

Microsoft uses the information to confirm a properly licensed copy of the software. Microsoft doesn't use the information to contact individual consumers.

For more information, see Microsoft Privacy Statement .

Feedback

Was this page helpful? Yes No

Provide product feedback

Volume Activation Management Tool (VAMT) technical reference

The Volume Activation Management Tool (VAMT) allows automation and central management of the retail-activation process for Windows, Office, and select other Microsoft products. VAMT can manage volume activation using Multiple Activation Keys (MAKs) or the Windows Key Management Service (KMS). VAMT is a standard Microsoft Management Console (MMC) snap-in. VAMT can be installed on any computer that has a supported Windows OS version.

> **Important**
>
> VAMT is designed to manage volume activation for all currently supported versions of Windows, Windows Server, and Office.

VAMT is only available in an EN-US (x86) package.

Next steps

Introduction to VAMT

Last updated on 05/14/2025

Introduction to VAMT

The Volume Activation Management Tool (VAMT) enables network administrators and other IT professionals to automate and centrally manage the Windows, Office, and select other Microsoft products volume and retail activation process. VAMT can manage volume activation using Multiple Activation Keys (MAKs) or the Windows Key Management Service (KMS). VAMT is a standard Microsoft Management Console (MMC) snap-in and can be installed on any computer that has a supported Windows OS version.

> **Note**
>
> VAMT can be installed on, and can manage, physical or virtual instances. VAMT can't detect whether or not the remote products are virtual. As long as the products can respond to Windows Management Instrumentation (WMI) calls, they will be discovered and activated.

Managing MAK and retail activation

You can use a MAK or a retail product key to activate Windows, Windows Server, or Office on an individual computer or a group of computers. VAMT enables two different activation scenarios:

- **Online activation**: Many organizations maintain a single Windows system image or Office installation package for deployment across the organization. Occasionally there's also a need to use retail product keys in special situations. Online activation enables you to activate over the internet any products installed with MAK, KMS host, or retail product keys on one or more connected computers within a network. This process requires that each product communicate activation information directly to Microsoft.

- **Proxy activation**: This activation method enables you to perform volume activation for products installed on client computers that don't have internet access. The VAMT host computer distributes a MAK, KMS host key (CSVLK), or retail product key to one or more client products and collects the installation ID (IID) from each client product. The VAMT host sends the IIDs to Microsoft on behalf of the client products and obtains the corresponding Confirmation IDs (CIDs). The VAMT host then installs the CIDs on the client products to complete the activation. Using this method, only the VAMT host computer needs internet access. You can also activate products installed on computers in a workgroup that's isolated from any larger network, by installing a second instance of VAMT on a computer within the workgroup. Then, use removable media to transfer

activation data between this new instance of VAMT and the internet-connected VAMT host.

Managing KMS activation

In addition to MAK or retail activation, you can use VAMT to perform volume activation using the KMS. VAMT can install and activate GVLK (KMS client) keys on client products. GVLKs are the default product keys used by volume license editions of Windows, Windows Server, and Office.

VAMT treats a KMS host key (CSVLK) product key identically to a retail-type product key. The experience for product key entry and activation management are identical for both these product key types.

Enterprise environment

VAMT is commonly implemented in enterprise environments. The following screenshot illustrates three common environments: core network, secure zone, and isolated lab.

- In the core network environment, all computers are within a common network managed by Active Directory Domain Services (AD DS).
- The secure zone represents higher-security core network computers that have extra firewall protection.
- The isolated lab environment is a workgroup that is physically separate from the core network, and its computers don't have internet access. The network security policy states that no information that could identify a specific computer or user may be transferred out of the isolated lab.

VAMT user interface

The following screenshot shows the VAMT graphical user interface:

VAMT provides a single, graphical user interface for managing activations, and for performing other activation-related tasks such as:

- **Adding and removing computers**: You can use VAMT to discover computers in the local environment. VAMT can discover computers by querying AD DS, workgroups, by individual computer name or IP address, or via a general LDAP query.
- **Discovering products**: You can use VAMT to discover Windows, Windows Server, Office, and select other products installed on the client computers.

- **Monitoring activation status**: You can collect activation information about each product, including the last five characters of the product key being used, the current license state (such as Licensed, Grace, Unlicensed), and the product edition information.
- **Managing product keys**: You can store multiple product keys and use VAMT to install these keys to remote client products. You can also determine the number of activations remaining for MAKs.
- **Managing activation data**: VAMT stores activation data in an SQL database. VAMT can export this data to other VAMT hosts or to an archive in XML format.

Next steps

VAMT requirements

Last updated on 05/14/2025

Active Directory-Based Activation overview

Active Directory-Based Activation (ADBA) enables enterprises to activate computers through a connection to their domain. Many companies have computers at offsite locations that use products that are registered to the company. Previously these computers needed to either use a retail key or a Multiple Activation Key (MAK), or physically connect to the network in order to activate their products by using Key Management Services (KMS). ADBA provides a way to activate these products if the computers can join the company's domain. When the user joins their computer to the domain, the ADBA object automatically activates Windows installed on their computer, as long as the computer has a Generic Volume License Key (GVLK) installed. No single physical computer is required to act as the activation object, because it's distributed throughout the domain. ADBA has the following benefits

- No additional devices required to maintain (KMS host)
- No RPC requirement, uses LDAP
- Works with RODC

Note: ADBA activation only works in single forest, even if you have trust relationship setup. Each forest requires it's own ADBA object. KMS can work cross forest if the DNS SRV record is manually created under the DNS of each forest, provided 2-way trust relationship is created

ADBA scenarios

You might use ADBA if you only want to activate domain joined devices.

If you have a server hosting the KMS service, it can be necessary to reactivate licenses if the server is replaced with a new host. Reactivating licenses isn't necessary When ADBA is used.

ADBA can also make load balancing easier when multiple KMS servers are present since the client can connect to any domain controller. ADBA is simpler than using the DNS service to load balance by configuring priority and weight values.

Some VDI solutions also require that new clients activate during creation before they're added to the pool. In this VDI scenario, ADBA can eliminate potential VDI issues that might arise due to a KMS outage.

ADBA methods

VAMT enables IT Professionals to manage and activate the ADBA object. Activation can be performed using the following methods:

- Online activation: To activate an ADBA forest online, the user selects the **Online activate forest** function, selects a KMS Host key (CSVLK) to use, and gives the ADBA Object a name.
- Proxy activation: For a proxy activation, the user first selects the **Proxy activate forest** function, selects a KMS Host key (CSVLK) to use, gives the ADBA Object a name, and provides a file name to save the CILx file that contains the Installation ID. Next, the user takes that file to a computer that is running VAMT with an Internet connection and then selects the **Acquire confirmation IDs for CILX** function on the VAMT landing page, and provides the original CILx file. When VAMT has loaded the Confirmation IDs into the original CILx file, the user takes this file back to the original VAMT instance, where the user completes the proxy activation process by selecting the **Apply confirmation ID to Active Directory domain** function.

Related articles

- How to Activate an Active Directory Forest Online
- How to Proxy Activate an Active Directory Forest

Last updated on 06/10/2025

VAMT requirements

This article includes info about the product key and system requirements for VAMT.

Product key requirements

The Volume Activation Management Tool (VAMT) can be used to perform activations using any of the following types of product keys.

Expand table

Product key type	Where to obtain
• Multiple Activation Key (MAK) • Key Management Service (KMS) host key (CSVLK) • KMS client setup keys (GVLK)	Volume licensing keys can only be obtained with a signed contract from Microsoft. For more information, see the Microsoft Volume Licensing portal .
Retail product keys	Obtained at time of product purchase.

System requirements

The following table lists the system requirements for the VAMT host computer.

Expand table

Item	Minimum system requirement
Computer and Processor	1 GHz x86 or x64 processor
Memory	1 GB RAM for x86 or 2 GB RAM for x64
Hard Disk	16 GB available hard disk space for x86 or 20 GB for x64
External Drive	Removable media (Optional)
Display	1024x768 or higher resolution monitor
Network	Connectivity to remote computers via Windows Management Instrumentation (TCP/IP) and Microsoft Activation Web Service on the Internet via HTTPS
Operating System	Currently supported versions of Windows client and Windows Server.

Item	Minimum system requirement
Additional Requirements	• Connection to a SQL Server database. For more info, see Install VAMT. • PowerShell, which is included with all currently supported versions of Windows.

Related content

Install VAMT

Configure client computers

Last updated on 05/14/2025

Install VAMT

Applies to: ✅ Windows 11, ✅ Windows 10, ✅ Windows Server 2022, ✅ Windows Server 2019, ✅ Windows Server 2016

This article describes how to install the Volume Activation Management Tool (VAMT). VAMT is installed as part of the Windows Assessment and Deployment Kit (ADK) for Windows.

> **Important**
>
> VAMT requires local administrator privileges on all managed computers in order to deposit confirmation IDs (CIDs), get the client products' license status, and install product keys. If VAMT is being used to manage products and product keys on the local host computer but administrator privileges aren't available, start VAMT with elevated privileges. For best results when using Active Directory-based activation, we recommend running VAMT while logged on as a domain administrator.

> **Note**
>
> The VAMT Microsoft Management Console snap-in ships as an x86 package.

Requirements

- Windows Server with Desktop Experience, with internet access (for the main VAMT console) and all updates applied.
- Latest version of the Windows ADK.
- Any supported SQL Server Express version. The latest is recommended.
- Alternatively, any supported **full** SQL instance.

Install SQL Server Express / alternatively use any full SQL instance

1. Download and open the SQL Server Express package.
2. For **Select an installation type:**, select **Basic**.

3. In the **Microsoft SQL Server Server License Terms** screen, accept the license terms by selecting the **Accept** button.

4. In the **Specify SQL Server install location** screen under **INSTALL LOCATION *:**, specify an install location or use the default path, and then select the **Install** button.

5. Once the installation is complete, in the **Installation Has completed successfully!** page, under **INSTANCE NAME**, note the instance name for the installation. The instance name is used later in the Configure VAMT to connect to SQL Server Express or full SQL Server section.

6. Once the instance name is noted, select the **Close** button, and then select the **Yes** button to confirm exiting the installer.

Install VAMT using the ADK

1. Download the latest version of Windows ADK.

   If an older version is already installed, it's recommended to first uninstall the older ADK before installing the latest version. Existing VAMT data is maintained in the VAMT database.

2. Open the ADK installer that was downloaded in the previous step. The **Windows Assessment and Deployment Kit** window opens.

3. In the **Windows Assessment and Deployment Kit** window:

   a. At the **Specify Location** page, under **Install Path:**, enter an install location or use the default path. It's recommended to install at the default path. Once done, select the **Next** button.

   b. In the **Windows Kits Privacy** page, select a privacy setting, and then select the **Next** button.

   c. In the **License Agreement** page, accept the license terms by selecting the **Accept** button.

   d. In the **Select the features you want to install** page, select **Volume Activation Management Tool (VAMT)**. If desired, select any additional features to install. Once done, select the **Install** button.

   e. Once installation is complete, the **Welcome to the Windows Assessment and Deployment Kit!** page is displayed. Select the **Close** button.

Configure VAMT to connect to SQL Server Express or full SQL Server

1. In the Start Menu under **Windows Kits**, select **Volume Active Management Tool 3.1**. The **Database Connection Settings** window opens.

2. In the **Database Connection Settings** window:

   a. Next to **Server:**, enter the server instance name as determined in the Install SQL Server Express / alternatively use any full SQL instance section. If SQL is remote, make sure to use the FQDN.

   b. Next to **Database:**, add a name for the database.

   c. Once the database server and database names are entered, select the **Connect** button.

   d. Select the **Yes** button to create the database.

Uninstall VAMT

To uninstall VAMT:

1. Right-click on the Start Menu and select **Settings**.

2. Select **Apps** in the left hand pane.

3. In the right hand pane under **Apps**, select **Installed apps**.

   Alternatively, select the following link to automatically open the **Settings** app to the **Installed apps** page:

   Installed apps

4. Scroll through the list of installed apps and find **Windows Assessment and Deployment Kit**.

5. Select the three dots **...** next to **Windows Assessment and Deployment Kit** and then select **Modify**. The **Windows Assessment and Deployment Kit** window opens.

6. In the **Windows Assessment and Deployment Kit** window:

   a. In the **Maintain your Windows Assessment and Deployment Kit features** page, select **Change**, and then select the **Next** button.

b. In the **Select the features you want to change** page, uncheck **Volume Activation Management Tool (VAMT)**, and then select the **Change** button.

c. Once the uninstall is complete, the **Change is complete.** page is displayed. Select the **Close** button.

Last updated on 05/14/2025

Configure client computers

To enable the Volume Activation Management Tool (VAMT) to function correctly, certain configuration changes are required on all client computers:

- An exception must be set in the client computer's firewall.
- A registry key must be created and set properly, for computers in a workgroup; otherwise, Windows® User Account Control (UAC) won't allow remote administrative operations.

Organizations where the VAMT will be widely used may benefit from making these changes inside the master image for Windows.

> **Important**
>
> This procedure only applies to clients running Windows Vista or later. For clients running Windows XP Service Pack 1, see Connecting Through Windows Firewall.

Configuring the Windows Firewall to allow VAMT access

Enable the VAMT to access client computers using the **Windows Firewall** Control Panel:

1. Open Control Panel and double-click **System and Security**.
2. Select **Windows Firewall**.
3. Select **Allow a program or feature through Windows Firewall**.
4. Select the **Change settings** option.
5. Select the **Windows Management Instrumentation (WMI)** checkbox.
6. Select **OK**.

> **Warning**
>
> By default, Windows Firewall Exceptions only apply to traffic originating on the local subnet. To expand the exception to apply to multiple subnets, you need to change the exception settings in the Windows Firewall with Advanced Security, as described below.

Configure Windows Firewall to allow VAMT access across multiple subnets

Enable the VAMT to access client computers across multiple subnets using the **Windows Firewall with Advanced Security** Control Panel:

1. Open the Control Panel and double-click **Administrative Tools**.
2. Select **Windows Firewall with Advanced Security**.
3. Make your changes for each of the following three WMI items, for the applicable Network Profile (Domain, Public, Private):
    - Windows Management Instrumentation (ASync-In)
    - Windows Management Instrumentation (DCOM-In)
    - Windows Management Instrumentation (WMI-In)
4. In the **Windows Firewall with Advanced Security** dialog box, select **Inbound Rules** from the left-hand panel.
5. Right-click the desired rule and select **Properties** to open the **Properties** dialog box.
    - On the **General** tab, select the **Allow the connection** checkbox.
    - On the **Scope** tab, change the Remote IP Address setting from "Local Subnet" (default) to allow the specific access you need.
    - On the **Advanced** tab, verify selection of all profiles that are applicable to the network (Domain or Private/Public).

    In certain scenarios, only a limited set of TCP/IP ports are allowed through a hardware firewall. Administrators must ensure that WMI (which relies on RPC over TCP/IP) is allowed through these types of firewalls. By default, the WMI port is a dynamically

allocated random port above 1024. The following Microsoft knowledge article discusses how administrators can limit the range of dynamically allocated ports. Limiting the range of dynamically allocated ports is useful if, for example, the hardware firewall only allows traffic in a certain range of ports.

For more info, see How to configure RPC dynamic port allocation to work with firewalls.

Create a registry value for the VAMT to access workgroup-joined computer

> ⚠ **Warning**
>
> This section contains information about how to modify the registry. Make sure to back up the registry before you modify it; in addition, ensure that you know how to restore the registry, if a problem occurs. For more information about how to back up, restore, and modify the registry, see **Windows registry information for advanced users**.

On the client computer, create the following registry key using regedit.exe.

1. Navigate to `HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\system`
2. Enter the following details:
   - **Value Name: LocalAccountTokenFilterPolicy**
   - **Type: DWORD**
   - **Value Data: 1**

   > ⓘ **Note**
   >
   > To discover VAMT-manageable Windows computers in workgroups, you must enable network discovery on each client.

Deployment options

There are several options for organizations to configure the WMI firewall exception for computers:

- **Image.** Add the configurations to the master Windows image deployed to all clients.

- **Group Policy.** If the clients are part of a domain, then all clients can be configured using Group Policy. The Group Policy setting for the WMI firewall exception is found in GPMC.MSC at: **Computer Configuration** > **Windows Settings** > **Security Settings** > **Windows Firewall with Advanced Security** > **Windows Firewall with Advanced Security** > **Inbound Rules**.
- **Script.** Execute a script using Microsoft Configuration Manager or a third-party remote script execution facility.
- **Manual.** Configure the WMI firewall exception individually on each client.

The above configurations will open an additional port through the Windows Firewall on target computers and should be performed on computers that are protected by a network firewall. In order to allow VAMT to query the up-to-date licensing status, the WMI exception must be maintained. We recommend administrators consult their network security policies and make clear decisions when creating the WMI exception.

Related articles

VAMT requirements

Install VAMT

Last updated on 05/14/2025

Add and remove computers

You can add computers that have any of the supported Windows or Office products installed to a Volume Activation Management Tool (VAMT) database by using the **Discover products** function. You can search for computers in an Active Directory domain, by individual computer name or IP address, in a workgroup, or by a general LDAP query. You can remove computers from a VAMT database by using the **Delete** function. After you add the computers, you can add the products that are installed on the computers by running the **Update license status** function.

Before adding computers, ensure that the Windows Management Instrumentation (WMI) firewall exception required by VAMT has been enabled on all target computers. For more information, see Configure Client Computers.

To add computers to a VAMT database

1. Open VAMT.

2. Select **Discover products** in the **Actions** menu in the right-side pane to open the **Discover Products** dialog box.

3. In the **Discover products** dialog box, select **Search for computers in the Active Directory** to display the search options, then select the search option you want to use. You can search for computers in an Active Directory domain, by individual computer name or IP address, in a workgroup, or by a general LDAP query.

    - To search for computers in an Active Directory domain, select **Search for computers in the Active Directory**, then under **Domain Filter Criteria**, in the list of domain names select the name of the domain you want to search. You can narrow the search further by typing a name in the **Filter by computer name** field to search for a specific computer within the domain. This filter supports the asterisk (*) wildcard. For example, typing "a*" will display only computer names that start with the letter "a".

    - To search by individual computer name or IP address, select **Manually enter name or IP address**, then enter the full name or IP address in the **One or more computer names or IP addresses separated by commas** text box. Separate multiple entries with a comma. VAMT supports both IPv4 and IPV6 addressing.

    - To search for computers in a workgroup, select **Search for computers in the workgroup**, then under **Workgroup Filter Criteria**, in the list of workgroup names select the name of the workgroup you want to search. You can narrow the search further by typing a name in the **Filter by computer name** field to search for a

specific computer within the workgroup. This filter supports the asterisk (*) wildcard. For example, typing "a*" will display only computer names that start with the letter "a".

- To search for computers by using a general LDAP query, select **Search with LDAP query** and enter your query in the text box provided. VAMT will validate only the LDAP query syntax, but will otherwise run the query without further checks.

4. Select **Search**.

5. VAMT searches for the specified computers and adds them to the VAMT database. During the search, VAMT displays the **Finding computers** message shown below.

   To cancel the search, select **Cancel**. When the search is complete, the names of the newly discovered computers appear in the product list view in the center pane.

> ⓘ **Important**
>
> This step adds only the computers to the VAMT database, and not the products that are installed on the computers. To add the products, you need to run the **Update license status** function.

To add products to VAMT

1. In the **Products** list, select the computers that need to have their product information added to the VAMT database.

2. You can use the **Filter** function to narrow your search for computers by clicking **Filter** in the right-side pane to open the **Filter Products** dialog box.

3. In the **Filter Products** dialog box, you can filter the list by computer name, product name, product key type, license status, or by any combination of these options.
    - To filter the list by computer name, enter a name in the **Computer Name** box.
    - To filter the list by Product Name, Product Key Type, or License Status, select the list you want to use for the filter and select an option. If necessary, select **clear all filters** to create a new filter.
4. Select **Filter**. VAMT displays the filtered list in the center pane.
5. In the right-side **Actions** pane, select **Update license status** and then select a credential option. Choose **Alternate Credentials** only if you're updating products that require administrator credentials different from the ones you used to log into the computer. If you're supplying alternate credentials, in the **Windows Security** dialog box type the appropriate user name and password and select **OK**.
6. VAMT displays the **Collecting product information** dialog box while it collects the licensing status of all supported products on the selected computers. When the process is finished, the updated licensing status of each product will appear in the product list view in the center pane.

    > **Note**

    If a computer has more than one supported product installed, VAMT adds an entry for each product. The entry appears under the appropriate product heading.

To remove computers from a VAMT database

You can delete a computer by clicking on it in the product list view, and then clicking **Delete** in the **Selected Item** menu in the right-hand pane. In the **Confirm Delete Selected Products** dialog box that appears, select **Yes** to delete the computer. If a computer has multiple products listed, you must delete each product to completely remove the computer from the VAMT database.

Related articles

Update product status

Remove products

Last updated on 05/14/2025

Update product status

After computers are added to the Volume Activation Management Tool (VAMT) database, the **Update license status** function needs to be used to add the products that are installed on the computers. The **Update license status** can also be used at any time to retrieve the most current license status for any products in the VAMT database. To retrieve license status, VAMT must have administrative permissions on all selected computers and Windows Management Instrumentation (WMI) must be accessible through the Windows Firewall. In addition, for workgroup computers, a registry key must be created to enable remote administrative actions under User Account Control (UAC). For more information, see Configure client computers.

> **Note**
>
> The license-status query requires a valid computer name for each system queried. If the VAMT database contains computers that were added without Personally Identifiable Information, computer names will not be available for those computers, and the status for these computers will not be updated.

Update the license status of a product

1. Open VAMT.
2. In the **Products** list, select one or more products that need to have their status updated.
3. In the right-side **Actions** pane, select **Update license status** and then select a credential option. Choose **Alternate Credentials** only if updating products that require administrator credentials different from the ones used to log into the computer.
4. If supplying alternate credentials, in the **Windows Security** dialog box type the appropriate user name and password and select **OK**.

   VAMT displays the **Collecting product information** dialog box while it collects the status of all selected products. When the process is finished, the updated licensing status of each product appears in the product list view in the center pane.

   > **Note**
   >
   > If a previously discovered Microsoft Office product is uninstalled from the remote computer, updating its licensing status causes the entry to be deleted from the **Office** product list view that results in the total number of discovered products being

smaller. However, the Windows installation of the same computer isn't deleted and is always be shown in the **Windows** products list view.

Related content

Add and remove computers

Remove products

Last updated on 05/14/2025

Remove products

To remove one or more products from the Volume Activation Management Tool (VAMT), you can delete them from the product list view in the center pane.

To delete one or more products

1. Select a product node in the left-side pane.
2. You can use the **Filter** function to narrow your search for computers by clicking **Filter** in the right-side pane to open the **Filter Products** dialog box.
3. In the **Filter Products** dialog box, you can filter the list by computer name, product name, product key type, license status, or by any combination of these options.
   - To filter the list by computer name, enter a name in the **Computer Name** box.
   - To filter the list by Product Name, Product Key Type, or License Status, select the list you want to use for the filter and select an option. If necessary, select **clear all filters** to create a new filter.
4. Select **Filter**. VAMT displays the filtered list in the center pane.
5. Select the products you want to delete.
6. Select **Delete** in the **Selected Items** menu in the right-side pane.
7. On the **Confirm Delete Selected Products** dialog box, select **OK**.

Related articles

Add and remove computers

Update product status

Last updated on 05/14/2025

Add and remove a product key

Before you can use a Multiple Activation Key (MAK), retail, or KMS Host key (CSVLK) product key, you must first add it to the Volume Activation Management Tool (VAMT) database.

To add a product key

1. Open VAMT.
2. In the left-side pane, right-click the **Product Keys** node to open the **Actions** menu.
3. Select **Add product keys** to open the **Add Product Keys** dialog box.
4. In the **Add Product Keys** dialog box, select from one of the following methods to add product keys:
   - To add product keys manually, select **Enter product key(s) separated by line breaks**, enter one or more product keys separated by line breaks, and select **Add Key(s)**.
   - To import a Comma Separated Values (CSV) file containing a list of product keys, select **Select a product key file to import**, browse to the file location, select **Open** to import the file, and then select **Add Key(s)**.

> **Note**
>
> If you are activating a large number of products with a MAK, you should refresh the activation count of the MAK, to ensure that the MAK can support the required number of activations. In the product key list in the center pane, select the MAK and click **Refresh product key data online** in the right-side pane to contact Microsoft and retrieve the number of remaining activations for the MAK. This step requires Internet access. You can only retrieve the remaining activation count for MAKs.

Remove a product key

- To remove a product key from the list, select the key in the list and select **Delete** on the **Selected Items** menu in the right-side pane. Select **Yes** to confirm deletion of the product key. Removing a product key from the VAMT database won't affect the activation state of any products or computers on the network.

Related articles

Install a product key

Install a KMS client key

Last updated on 05/14/2025

Install a product key

You can use the Volume Activation Management Tool (VAMT) to install retail, Multiple Activation Key (MAK), and KMS Host key (CSVLK).

To install a product key

1. Open VAMT.
2. In the left-side pane, select the product that you want to install keys onto.
3. You can use the **Filter** function to narrow your search for computers by clicking **Filter** in the right-side pane to open the **Filter Products** dialog box.
4. In the **Filter Products** dialog box, you can filter the list by computer name, product name, product key type, license status, or by any combination of these options.
    - To filter the list by computer name, enter a name in the **Computer Name** box.
    - To filter the list by Product Name, Product Key Type, or License Status, select the list you want to use for the filter and select an option. If necessary, select **clear all filters** to create a new filter.
5. Select **Filter**.
6. In the products list view in the center pane, sort the list if needed and then select the products that need to have keys installed. You can use the **CTRL** key or the **SHIFT** key to select more than one product.
7. Select **Install product key** in the **Selected Items** menu in the right-side pane to display the **Install Product Key** dialog box.
8. The **Select Product Key** dialog box displays the keys that are available to be installed. Under **Recommended MAKs**, VAMT might display one or more recommended MAK based on the selected products. You can select a recommended product key or a product key from the **All Product Keys** list. Use the scroll bar if you need to view the **Description** for each key. When you've selected the product key you want to install, select **Install Key**. Only one key can be installed at a time.
9. VAMT displays the **Installing product key** dialog box while it attempts to install the product key for the selected products. When the process is finished, the status appears in the **Action Status** column of the dialog box. Select **Close** to close the dialog box. You can also select the **Automatically close when done** check box when the dialog box appears.

The same status is shown under the **Status of Last Action** column in the product list view in the center pane.

> **Note**
>
> Product key installation will fail if VAMT finds mismatched key types or editions. VAMT will display the failure status and will continue the installation for the next product in the list. For more information on choosing the correct MAK or KMS Host key (CSVLK), see How to Choose the Right Volume License Key for Windows.

Related articles

Add and remove a product key

Install a KMS client key

Last updated on 05/14/2025

Install a KMS client key

You can use the Volume Activation Management Tool (VAMT) to install Generic Volume License Key (GVLK), or KMS client, product keys. For example, if you're converting a MAK-activated product to KMS activation.

> **Note**
>
> By default, volume license editions of Windows Vista, Windows® 7, Windows 8, Windows 10, Windows Server 2008, Windows Server 2008 R2, Windows Server® 2012, and Microsoft® Office 2010 use KMS for activation. GVLKs are already installed in volume license editions of these products.

To install a KMS client key

1. Open VAMT.
2. In the left-side pane, select **Products** to open the product list view in the center pane.
3. In the products list view in the center pane, select the products that need to have GVLKs installed. You can use the **Filter** function to narrow your search for computers by clicking **Filter** in the right-side pane to open the **Filter Products** dialog box.
4. In the **Filter Products** dialog box, you can filter the list by computer name, product name, product key type, license status, or by any combination of these options.
   - To filter the list by computer name, enter a name in the **Computer Name** box.
   - To filter the list by Product Name, Product Key Type, or License Status, select the list you want to use for the filter and select an option. If necessary, select **clear all filters** to create a new filter.
5. Select **Filter**. VAMT displays the filtered list in the center pane.
6. Select **Install product key** in the **Selected Items** menu in the right-side pane to display the **Install Product Key** dialog box.
7. The **Install Product Key** dialog box displays the keys that are available to be installed.
8. Select the **Automatically select an AD or KMS client key** option and then select **Install Key**.

VAMT displays the **Installing product key** dialog box while it attempts to install the product key for the selected products. When the process is finished, the status appears in the **Action Status** column of the dialog box. Select **Close** to close the dialog box. You can also select the **Automatically close when done** check box when the dialog box appears.

The same status is shown under the **Status of Last Action** column in the product list view in the center pane.

Related articles

Run KMS activation

Add and remove a product key

Install a product key

Last updated on 05/14/2025

Run online activation

You can use the Volume Activation Management Tool (VAMT) to enable client products to be activated over the internet. You can install the client products with any kind of product key that is eligible for online activation—Multiple Activation Key (MAK), retail, and Windows Key Management Services (KMS) host key.

Requirements

Before running online activation, ensure that the network and the VAMT installation meet the following requirements:

- VAMT is installed on a central computer that has network access to all client computers.
- Both the VAMT host and client computers have internet access.
- The products that you want to activate are added to VAMT.
- VAMT has administrative permissions on all computers that you intend to activate, and that Windows Management Instrumentation (WMI) can be accessed through the Windows firewall. For more information, see Configure client computers.

The product keys that are installed on the client products must have a sufficient number of remaining activations. If you're activating a MAK key, you can retrieve the remaining number of activations for that key by selecting the MAK in the product key list in the center pane and then clicking **Refresh product key data online** in the right-side pane. This action retrieves the number of remaining activations for the MAK from Microsoft. This step requires internet access and that the remaining activation count can only be retrieved for MAKs.

To run an online activation

1. Open VAMT.
2. In the products list view in the center pane, sort the list if necessary. You can use the **Filter** function to narrow your search for computers by clicking **Filter** in the right-side pane to open the **Filter Products** dialog box.
3. In the **Filter Products** dialog box, you can filter the list by computer name, product name, product key type, license status, or by any combination of these options.
    - To filter the list by computer name, enter a name in the **Computer Name** box.

- To filter the list by Product Name, Product Key Type, or License Status, select the list you want to use for the filter and select an option. If necessary, select **clear all filters** to create a new filter.

4. Select **Filter**. VAMT displays the filtered list in the center pane.

5. Select the products that you want to activate. You can use the **CTRL** key or the **SHIFT** key to select more than one product.

6. Select **Activate** in the **Selected Items** menu in the right-side **Actions** pane and then point to **Activate**. If the **Actions** pane isn't displayed, select the Show/Hide Action Pane button, which is located on the toolbar to the right of the Help button.

7. Point to **Online activate**, and then select the appropriate credential option. If you select the **Alternate Credentials** option, you'll be prompted to enter an alternate user name and password.

8. VAMT displays the **Activating products** dialog box until it completes the requested action. When activation is complete, the status appears in the **Action Status** column of the dialog box. Select **Close** to close the dialog box. You can also select the **Automatically close when done** check box when the dialog box appears.

   The same status is shown under the **Status of Last Action** column in the products list view in the center pane.

   > **Note**
   >
   > Online activation doesn't enable you to save the Confirmation IDs (CIDs). As a result, you can't run local reactivation.

   > **Note**
   >
   > You can use online activation to select products that have different key types and activate the products at the same time.

Related articles

Run proxy activation

Run KMS activation

Run local reactivation

Last updated on 05/14/2025

Run proxy activation

Article • 07/25/2024

You can use the Volume Activation Management Tool (VAMT) to run activation for client computers that don't have internet access. The client products can be installed with any type of product key that is eligible for proxy activation: Multiple activation Key (MAK), KMS Host key (CSVLK), or retail key.

In a typical proxy-activation scenario, the VAMT host computer distributes a MAK to one or more client computers and collects the installation ID (IID) from each computer. The VAMT host computer sends the IIDs to Microsoft on behalf of the client computers and obtains the corresponding Confirmation IDs (CIDs). The VAMT host computer then installs the CIDs on the client computer to complete the activation. Using this activation method, only the VAMT host computer needs internet access.

> **Note**
>
> For workgroups that are completely isolated from any larger network, you can still run MAK, KMS Host key (CSVLK), or retail proxy activation. This requires installing a second instance of VAMT on a computer within the isolated group and using removable media to transfer activation data between that computer and another VAMT host computer that has internet access. For more information about this scenario, see Scenario 2: Proxy Activation. Similarly, you can proxy activate a KMS Host key (CSVLK) located in an isolated network. You can also proxy activate a KMS Host key (CSVLK) in the core network if you do not want the KMS host computer to connect to Microsoft over the internet.

Requirements

Before running proxy activation, ensure that your network and the VAMT installation meet the following requirements:

- There's an instance of VAMT that is installed on a computer that has internet access. If you're running proxy activation for an isolated workgroup, you also need to have VAMT installed on one of the computers in the workgroup.
- The products to be activated have been added to VAMT and are installed with a retail product key, a KMS Host key (CSVLK) or a MAK. If the products haven't been installed with a proper product key, refer to the steps in the Add and Remove a Product Key section for instructions on how to install a product key.

- VAMT has administrative permissions on all products to be activated and Windows Management Instrumentation (WMI) is accessible through the Windows firewall.
- For workgroup computers, a registry key must be created to enable remote administrative actions under User Account Control (UAC). For more information, see Configure client computers.

  The product keys that are installed on the client products must have a sufficient number of remaining activations. If you're activating a MAK key, you can retrieve the remaining number of activations for that key by selecting the MAK in the product key list in the center pane and then clicking **Refresh product key data online** in the right-side pane. This action retrieves the number of remaining activations for the MAK from Microsoft. This step requires internet access and that the remaining activation count can only be retrieved for MAKs.

To run proxy activation

1. Open VAMT.
2. If necessary, install product keys. For more information, see:
   - Install a product key to install retail, MAK, or KMS Host key (CSVLK).
   - Install a KMS Client Key to install GVLK (KMS client) keys.
3. In the **Products** list in the center pane, select the individual products to be activated. You can use the **Filter** function to narrow your search for products by clicking **Filter** in the right-side pane to open the **Filter Products** dialog box.
4. In the **Filter Products** dialog box, you can filter the list by computer name, product name, product key type, license status, or by any combination of these options.
   - To filter the list by computer name, enter a name in the **Computer Name** box.
   - To filter the list by Product Name, Product Key Type, or License Status, select the list you want to use for the filter and select an option. If necessary, select **clear all filters** to create a new filter.
5. Select **Filter**. VAMT displays the filtered list in the center pane.
6. In the right-side pane, select **Activate** and then select **Proxy activate** to open the **Proxy Activate** dialog box.
7. In the **Proxy Activate** dialog box select **Apply Confirmation ID, apply to selected machine(s) and activate**.

8. If you're activating products that require administrator credentials different from the ones you're currently using, select the **Use Alternate Credentials** checkbox.

9. Select **OK**.

10. VAMT displays the **Activating products** dialog box until it completes the requested action. If you selected the **Alternate Credentials** option, you'll be prompted to enter the credentials.

    > **Note**
    >
    > You can use proxy activation to select products that have different key types and activate the products at the same time.

Related articles

Run online activation

Run KMS activation

Run local reactivation

Feedback

Was this page helpful? Yes No

Provide product feedback

Run KMS activation

The Volume Activation Management Tool (VAMT) can be used to run volume activation using the Key Management Service (KMS). You can use VAMT to activate Generic Volume Licensing Keys, or KMS client keys, on products accessible to VAMT. GVLKs are the default product keys used by the volume-license editions of Windows Vista, Windows 7, Windows 8, Windows 10, Windows Server 2008, Windows Server 2008 R2, Windows Server® 2012, and Microsoft Office 2010. GVLKs are already installed in volume-license editions of these products.

Requirements

Before configuring KMS activation, ensure that your network and VAMT installation meet the following requirements:

- KMS host is set up and enabled.
- KMS clients can access the KMS host.
- VAMT is installed on a central computer with network access to all client computers.
- The products to be activated have been added to VAMT. For more information on adding product keys, see Install a KMS Client Key.
- VAMT has administrative permissions on all computers to be activated, and Windows Management Instrumentation (WMI) is accessible through the Windows Firewall. For more information, see Configure client computers.

To configure devices for KMS activation

1. Open VAMT.
2. If necessary, set up the KMS activation preferences. If you don't need to set up the preferences, skip to step 6 in this procedure. Otherwise, continue to step 2.
3. To set up the preferences, on the menu bar select **View**, then select **Preferences** to open the **Volume Activation Management Tool Preferences** dialog box.
4. Under **Key Management Services host selection**, select one of the following options:
    - **Find a KMS host automatically using DNS (default)**. If you choose this option, VAMT first clears any previously configured KMS host on the target computer, and instructs the computer to query the Domain Name Service (DNS) to locate a KMS host and attempt activation.

- **Find a KMS host using DNS in this domain for supported products**. Enter the domain name. If you choose this option, VAMT first clears any previously configured KMS host on the target computer, and instructs the computer to query the DNS in the specified domain to locate a KMS host and attempt activation.
- **Use specific KMS host**. Enter the KMS host name and KMS host port. For environments that don't use DNS for KMS host identification, VAMT sets the specified KMS host name and KMS host port on the target computer, and then instructs the computer to attempt activation with the specific KMS host.

5. Select **Apply**, and then select **OK** to close the **Volume Activation Management Tool Preferences** dialog box.
6. Select the products to be activated by selecting individual products in the product list view in the center pane. You can use the **Filter** function to narrow your search for computers by clicking **Filter** in the right-side pane to open the **Filter Products** dialog box. In the **Filter Products** dialog box, you can filter the list by computer name, product name, product key type, license status, or by any combination of these options.
   - To filter the list by computer name, enter a name in the **Computer Name** box.
   - To filter the list by Product Name, Product Key Type, or License Status, select the list you want to use for the filter and select an option. If necessary, select **clear all filters** to create a new filter.
7. Select **Filter**. VAMT displays the filtered list in the center pane.
8. In the right-side pane, select **Activate** in the **Selected Items** menu, and then select **Volume activate**.
9. Select a credential option. Choose **Alternate credentials** only if you're activating products that require administrator credentials different from the ones you're currently using.
10. If you're supplying alternate credentials, at the prompt, type the appropriate user name and password and select **OK**. VAMT displays the **Volume Activation** dialog box until it completes the requested action. When the process is finished, the updated activation status of each product appears in the product list view in the center pane.

Related articles

Run online activation

Run proxy activation

Run local reactivation

Last updated on 05/14/2025

Run local reactivation

If you reinstall Windows® or Microsoft® Office 2010 on a computer that was initially activated using proxy activation (MAK, retail, or CSLVK (KMS host)), and have not made significant changes to the hardware, use this local reactivation procedure to reactivate the program on that computer. Local reactivation relies upon data that was created during the initial proxy activation and stored in the Volume Activation Management Tool (VAMT) database. The database contains the installation ID (IID) and confirmation ID (Pending CID). Local reactivation uses this data to reapply the CID and reactivate those products. Reapplying the same CID conserves the remaining activations on the key.

> **Note**
>
> During the initial proxy activation, the CID is bound to a digital "fingerprint", which is calculated from values assigned to several different hardware components in the computer. If the computer has had significant hardware changes, this fingerprint will no longer match the CID. In this case, you must obtain a new CID for the computer from Microsoft.

To run a local reactivation

1. Open VAMT. Make sure that you're connected to the desired database.
2. In the left-side pane, select the product you want to reactivate to display the products list.
3. In the product list view in the center pane, select the desired products to be reactivated. You can sort the list by computer name by clicking on the **Computer Name** heading. You can also use the **Filter** function to narrow your search for computers by clicking **Filter** in the right-side pane to open the **Filter Products** dialog box.
4. In the **Filter Products** dialog box, you can filter the list by computer name, product name, product key type, license status, or by any combination of these options.
    - To filter the list by computer name, enter a name in the **Computer Name** box.
    - To filter the list by Product Name, Product Key Type, or License Status, select the list you want to use for the filter and select an option. If necessary, select **clear all filters** to create a new filter.
5. Select **Filter**. VAMT displays the filtered list in the center pane.
6. In the right-side pane, select **Activate**, and then select **Apply Confirmation ID**.

7. Select a credential option. Choose **Alternate credentials** only if you're reactivating products that require administrator credentials different from the ones you're currently using.

8. If you're supplying alternate credentials, in the **Windows Security** dialog box type the appropriate user name, and password and select **OK**.

   VAMT displays the **Apply Confirmation ID** dialog box.

9. If you're using a different product key than the product key used for initial activation, you must complete a new activation to obtain a new CID.

10. If you're activating a product that requires administrator credentials different from the ones you're currently using, select the **Use Alternate Credentials** check box.

11. Select **OK**.

Related article

Run online activation

Run proxy activation

Run KMS activation

Last updated on 05/14/2025

Activate an Active Directory forest online

You can use the Volume Activation Management Tool (VAMT) Active Directory-Based Activation (ADBA) function to activate an Active Directory (AD) forest over the Internet. ADBA enables certain products to inherit activation from the domain.

> **Important**
>
> ADBA is only applicable to Generic Volume License Keys (GVLKs) and KMS Host keys (CSVLKs). To use ADBA, one or more KMS Host keys (CSVLKs) must be installed on the AD forest, and client keys (GVLKs) must be installed on the client products.

Requirements

Before performing online activation, ensure that the network and the VAMT installation meet the following requirements:

- VAMT is installed on a host computer that has Internet access.
- VAMT has administrative permissions to the Active Directory domain.
- The KMS Host key (CSVLK) you intend to use is added to VAMT in the **Product Keys** node.

To perform an online Active Directory forest activation

1. Open VAMT.
2. In the left-side pane, select the **Active Directory-Based Activation** node.
3. In the right-side **Actions** pane, select **Online activate forest** to open the **Install Product Key** dialog box.
4. In the **Install Product Key** dialog box, select the KMS Host key (CSVLK) that you want to apply to the AD forest.
5. If necessary, enter a new Active Directory-Based Activation Object name.

   > **Important**
   >
   > If you want to rename the ADBA object, you must do it now. After you click **Install Key**, the name cannot be changed.

6. Select **Install Key**.
7. VAMT displays the **Activating Active Directory** dialog box until it completes the requested action.

The activated object and the date that it was created appear in the **Active Directory-Based Activation** node in the center pane.

Related articles

- Scenario 1: online activation
- Add and remove computers

Last updated on 05/14/2025

Activate by proxy an Active Directory forest

You can use the Volume Activation Management Tool (VAMT) Active Directory-Based Activation (ADBA) function to activate by proxy an Active Directory (AD) forest for an isolated workgroup that doesn't have Internet access. ADBA enables certain volume products to inherit activation from the domain.

> **Important**
>
> ADBA is only applicable to *Generic Volume License Keys (GVLKs)* and *KMS Host key (CSVLK)*. To use ADBA, one or more KMS Host keys (CSVLK) must be installed on the AD forest, and client keys (GVLKs) must be installed on the client products.

In a typical proxy-activation scenario, the VAMT host computer distributes a product key to one or more client computers and collects the installation ID (IID) from each computer. The VAMT host computer sends the IIDs to Microsoft on behalf of the client computers and obtains the corresponding Confirmation IDs (CIDs). The VAMT host computer then installs the CIDs on the client computer to complete the activation. If you use this activation method, only the VAMT host computer needs to have Internet access.

> **Note**
>
> For workgroups that are isolated from any larger network, you can still perform an AD forest activation. This requires installing a second instance of VAMT on a computer in the isolated group and using removable media to transfer activation data between that computer and another VAMT host computer that has Internet access. You can also activate by proxy a KMS Host key (CSVLK) in the core network if you do not want the host computer to connect to Microsoft over the Internet.

Requirements

Before performing proxy activation, ensure that the network and the VAMT installation meet the following requirements:

- There's an instance of VAMT that is installed on a computer that has Internet access. If you're performing proxy activation for an isolated workgroup, you must also have VAMT installed on one of the computers in the workgroup.
- VAMT has administrative permissions to the Active Directory domain.

To perform an Active Directory forest proxy activation

1. Open VAMT.
2. In the left-side pane, select the **Active Directory-Based Activation** node.
3. In the right-side **Actions** pane, select **Proxy activate forest** to open the **Install Product Key** dialog box.
4. In the **Install Product Key** dialog box, select the KMS Host key (CSVLK) that you want to activate.
5. If you want to rename the ADBA object, enter a new Active Directory-Based Activation Object name. If you want to rename the ADBA object, you must do it now. After you select **Install Key**, the name can't be changed.
6. Enter the name of the file where you want to save the offline installation ID, or browse to the file location and then select **Open**. If you're activating an AD forest in an isolated workgroup, save the `.cilx` file to a removable media device.
7. Select **Install Key**. VAMT displays the **Activating Active Directory** dialog box until it completes the requested action. The activated object and the date that it was created appear in the **Active Directory-Based Activation** node in the center pane.
8. Insert the removable media into the VAMT host that has Internet access. Make sure that you are on the root node, and that the **Volume Activation Management Tool** view is displayed in the center pane.
9. In the right-side **Actions** pane, select **Acquire confirmation IDs for CILX** to open the **Acquire confirmation IDs for file** dialog box.
10. In the **Acquire confirmation IDs for file** dialog box, browse to where the `.cilx` file you exported from the isolated workgroup host computer is located. Select the file, and then select **Open**. VAMT displays an **Acquiring Confirmation IDs** message while it contacts Microsoft and acquires the CIDs.
11. When the CID collection process is complete, VAMT displays a **Volume Activation Management Tool** message that shows how many confirmation IDs were successfully acquired, and the name of the file to which the IDs were saved. Select **OK** to close the message.
12. Remove the storage device that contains the `.cilx` file from the Internet-connected VAMT host computer and insert it into the VAMT host computer in the isolated workgroup.

13. Open VAMT and then select the **Active Directory-Based Activation** node in the left-side pane.

14. In the right-side **Actions** pane, select **Apply confirmation ID to Active Directory domain**, browse to the `.cilx` file and then select **Open**.

VAMT displays the **Activating Active Directory** dialog box until it completes the requested action. The activated object and the date that it was created appear in the **Active Directory-Based Activation** node in the center pane.

Related articles

Add and remove computers

Last updated on 05/14/2025

Import and export Volume Activation Management Tool data

You can use the Volume Activation Management Tool (VAMT) to import product-activation data from a computer information list (`.cilx` or `.cil`) file into SQL Server. Also use VAMT to export product-activation data into a `.cilx` file. A `.cilx` file is an XML file that stores computer and product-activation data.

You can import data or export data during the following scenarios:

- Import and merge data from previous versions of VAMT.
- Export data to perform proxy activations.

> ⚠ **Warning**
>
> Editing a `.cilx` file through an application other than VAMT can corrupt the `.cilx` file. This method isn't supported.

Import VAMT data

To import data into VAMT, use the following process:

1. Open VAMT.
2. In the right-side **Actions** pane, select **Import list** to open the **Import List** dialog box.
3. In the **Import List** dialog box, navigate to the `.cilx` file location, choose the file, and select **Open**.
4. In the **Volume Activation Management Tool** dialog box, select **OK** to begin the import. VAMT displays a progress message while the file is being imported. Select **OK** when a message appears and confirms that the import has completed successfully.

Export VAMT data

Exporting VAMT data from a VAMT host computer that's not internet-connected is the first step of proxy activation using multiple VAMT hosts. To export product-activation data to a `.cilx` file:

1. In the left-side pane, select a product you want to export data for, or select **Products** if the list contains data for all products.
2. If you want to export only part of the data in a product list, in the product-list view in the center pane, select the products you want to export.
3. In the right-side **Actions** pane on, select **Export list** to open the **Export List** dialog box.
4. In the **Export List** dialog box, select **Browse** to navigate to the `.cilx` file.
5. Under **Export options**, select one of the following data-type options:
    - Export products and product keys
    - Export products only
    - Export proxy activation data only. Selecting this option makes sure that the export contains only the licensing information required for the proxy web service to obtain CIDs from Microsoft. No personally identifiable information (PII) is contained in the exported `.cilx` file when this selection is checked.
6. If you've selected products to export, select the **Export selected product rows only** check box.
7. Select **Save**. VAMT displays a progress message while the data is being exported. Select **OK** when a message appears and confirms that the export has completed successfully.

Related articles

VAMT proxy activation

Last updated on 05/14/2025

Use VAMT in Windows PowerShell

Article • 07/25/2024

The Volume Activation Management Tool (VAMT) PowerShell cmdlets can be used to perform the same functions as the Vamt.exe command-line tool.

Configuring VAMT in Windows PowerShell

Install PowerShell 3.0

VAMT PowerShell cmdlets require Windows PowerShell, which is included in Windows 10 and later. To download the latest version of PowerShell, see Installing PowerShell on Windows.

Install the Windows Assessment and Deployment Kit

In addition to PowerShell, you must import the VAMT PowerShell module. The module is included in the VAMT 3.0 folder after you install the Windows Assessment and Deployment Kit (Windows ADK).

Prepare the VAMT PowerShell environment

To open PowerShell with administrative credentials, select **Start** and enter `PowerShell` to locate the program. Right-click **Windows PowerShell**, and then select **Run as administrator**. To open PowerShell in Windows 7, select **Start**, select **All Programs**, select **Accessories**, select **Windows PowerShell**, right-click **Windows PowerShell**, and then select **Run as administrator**.

> **Important**
>
> If you are using a computer that has an 64-bit processor, select **Windows PowerShell (x86)**. VAMT PowerShell cmdlets are only supported for x86 architecture. You must use an x86 version of Windows PowerShell to import the VAMT module

The x86 versions of Windows PowerShell are available in the following directories:

- PowerShell: `C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe`
- PowerShell ISE: `C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell_ise.exe`

For all supported operating systems, you can use the VAMT PowerShell module included with the Windows ADK. By default, the module is installed with the Windows ADK in the VAMT folder. Change directories to the directory where VAMT is located. For example, if the Windows ADK is installed in the default location of `C:\Program Files(x86)\Windows Kits\10`, enter:

PowerShell

```
cd "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\VAMT3"
```

Import the VAMT PowerShell module

To import the VAMT PowerShell module, enter the following command at a PowerShell command prompt:

PowerShell

```
Import-Module .\VAMT.psd1
```

where **Import-Module** imports a module only into the current session. To import the module into all sessions, add an **Import-Module** command to a Windows PowerShell profile. For more information about profiles, enter `get-help about_profiles`.

To get help for VAMT PowerShell cmdlets

You can view all of the help sections for a VAMT PowerShell cmdlet, or you can view only the section that you're interested in. To view all of the Help content for a VAMT cmdlet, enter:

PowerShell

```
get-help <cmdlet name> -all
```

For example, enter:

PowerShell

```
get-help get-VamtProduct -all
```

⚠ **Warning**

The update-help cmdlet is not supported for VAMT PowerShell cmdlets. To view online help for VAMT cmdlets, you can use the `-online` option with the `get-help` cmdlet. For more information, see Volume Activation Management Tool (VAMT) Cmdlets in Windows PowerShell.

View VAMT PowerShell help sections

1. To get the syntax to use with a cmdlet, enter the following command at a PowerShell command prompt:

   PowerShell
   ```
   get-help <cmdlet name>
   ```

   For example, enter:

   PowerShell
   ```
   get-help get-VamtProduct
   ```

2. To see examples using a cmdlet, enter:

   PowerShell
   ```
   get-help <cmdlet name> -examples
   ```

   For example, enter:

   PowerShell
   ```
   get-help get-VamtProduct -examples
   ```

Feedback

Was this page helpful? Yes No

Provide product feedback

Scenario 1: online activation

Article • 02/28/2023

In this scenario, the Volume Activation Management Tool (VAMT) is deployed in the Core Network environment. VAMT is installed on a central computer that has network access to all of the client computers. Both the VAMT host and the client computers have Internet access. The following illustration shows a diagram of an online activation scenario for Multiple Activation Keys (MAKs). You can use this scenario for online activation of the following key types:

- Multiple Activation Key (MAK)
- Windows Key Management Service (KMS) keys:
  - KMS Host key (CSVLK)
  - Generic Volume License Key (GVLK), or KMS client key
- Retail The Secure Zone represents higher-security Core Network computers that have additional firewall protection.

Step 1: Install and start VAMT on a networked host computer

1. Install VAMT on the host computer.
2. Select the VAMT icon in the **Start** menu to open VAMT.

Step 2: Configure the Windows Management Instrumentation firewall exception on target computers

- Ensure that the Windows Management Instrumentation (WMI) firewall exception has been enabled for all target computers. For more information, see Configure client computers.

> **Note**
>
> To retrieve product license status, VAMT must have administrative permissions on the remote computers and WMI must be available through the Windows Firewall. In addition, for workgroup computers, a registry key must be created to enable remote administrative actions under User Account Control (UAC). For more information, see Configure client computers.

Step 3: Connect to a VAMT database

1. If you aren't already connected to a database, the **Database Connection Settings** dialog box appears when you open VAMT. Select the server and database where the keys that must be activated are located.
2. Select **Connect**.
3. If you're already connected to a database, VAMT displays an inventory of the products and product keys in the center pane, and a license overview of the computers in the database. If you need to connect to a different database, select **Successfully connected to Server** to open **the Database Connection Settings** dialog box. For more information about how to create VAMT databases and adding VAMT data, see Import and export VAMT data.

Step 4: Discover products

1. In the left-side pane, in the **Products** node Products, select the product that you want to activate.

2. To open the **Discover Products** dialog box, select **Discover products** in the **Actions** menu in the right-side pane.

3. In the **Discover Products** dialog box, select **Search for computers in the Active Directory** to display the search options, and then select the search options that you want to use. You can search for computers in an Active Directory domain, by individual computer name or IP address, in a workgroup, or by a general Lightweight Directory Access Protocol (LDAP) query:

   - To search for computers in an Active Directory domain, select **Search for computers in the Active Directory**. Then under **Domain Filter Criteria**, in the list of domain names select the name of the domain that you want to search. You can narrow the search further by typing a name in the **Filter by computer name** field to search for specific computers in the domain. This filter supports the asterisk (`*`) wildcard. For example, typing `a*` will display only those computer names that start with the letter "a".

   - To search by individual computer name or IP address, select **Manually enter name or IP address**. Then enter the full name or IP address in the **One or more computer names or IP addresses separated by commas** text box. Separate multiple entries with a comma. VAMT supports both IPv4 and IPV6 addressing.

   - To search for computers in a workgroup, select **Search for computers in the workgroup**. Then under **Workgroup Filter Criteria**, in the list of workgroup names, select the name of the workgroup that you want to search. You can narrow the search further by typing a name in the **Filter by computer name** field to search for a specific computer in the workgroup. This filter supports the asterisk (`*`) wildcard. For example, typing `a*` will display only computer names that start with the letter "a".

   - To search for computers by using a general LDAP query, select **Search with LDAP query** and enter your query in the text box that appears. VAMT will validate the LDAP query syntax, but will otherwise run the query without additional checks.

4. Select **Search**.

When the search is complete, the products that VAMT discovers appear in the product list view in the center pane.

Step 5: Sort and filter the list of computers

You can sort the list of products so that it's easier to find the computers that require product keys to be activated:

1. On the menu bar at the top of the center pane, select **Group by**, and then select **Product**, **Product Key Type**, or **License Status**.

2. To sort the list further, you can select one of the column headings to sort by that column.

3. You can also use the **Filter** function to narrow your search for computers by clicking **Filter** in the right-side pane to open the **Filter Products** dialog box.

4. In the **Filter Products** dialog box, you can filter the list by computer name, product name, product key type, license status, or by any combination of these options.

    - To filter the list by computer name, enter a name in the **Computer Name** box.

    - To filter the list by product name, product key type, or license status, select the list you want to use for the filter and select an option. If necessary, select **clear all filters** to create a new filter.

5. Select **Filter**. VAMT displays the filtered list in the product list view in the center pane.

Step 6: Collect status information from the computers in the list

To collect the status from select computers in the database, you can select computers in the product list view by using one of the following methods:

- To select a block of consecutively listed computers, select the first computer that you want to select, and then select the last computer while pressing the **Shift** key.

- To select computers that aren't listed consecutively, hold down the **Ctrl** key and select each computer for which you want to collect the status information.

To collect status information from the selected computers

- In the right-side **Actions** pane, select **Update license status** in the **Selected Items** menu and then select a credential option. Choose **Alternate Credentials** only if you're updating products that require administrator credentials that are different from the ones that you used to sign into the computer. Otherwise, select **Current Credentials** and continue to step 2. If you're supplying alternate credentials, in the **Windows Security** dialog box, type the appropriate user name and password and then select **OK**.
- VAMT displays the **Collecting product information** dialog box while it collects the license status of all supported products on the selected computers. When the process is finished, the updated license status of each product will appear in the product list view in the center pane.

> ⓘ **Note**
>
> If a computer has more than one supported product installed, VAMT adds an entry for each product. The entry appears under the appropriate product heading.

Step 7: Add product keys and determine the remaining activation count

1. Select the **Product Keys** node in the left-side pane, and then select **Add Product Keys** in the right-side pane to open the **Add Product Keys** dialog box.
2. In the **Add Product Key** dialog box, you can select from one of the following methods to add product keys:
   - To add product keys manually, select **Enter product key(s) separated by line breaks**, enter one or more product keys, and then select **Add Key(s)**.
   - To import a Comma Separated Values File (CSV) that contains a list of product keys, select **Select a product key file to import**, browse to the file location, select **Open** to import the file, and then select **Add Key(s)**.

   The keys that you have added appear in the **Product Keys** list view in the center pane.

> ⓘ **Important**

> If you are activating many products with a MAK, refresh the activation count of the MAK to ensure that the MAK can support the required number of activations. In the product key list in the center pane, select the MAK and then click **Refresh product key data online** in the right-side pane to contact Microsoft and retrieve the number of remaining activations for the MAK. This step requires Internet access. You can only retrieve the remaining activation count for MAKs.

Step 8: Install the product keys

1. In the left-side pane, select the product that you want to install keys on to.
2. If necessary, sort and filter the list of products so that it's easier to find the computers that must have a product key installed. See Step 5: Sort and filter the list of computers.
3. In the **Products** list view pane, select the individual products that must have keys installed. You can use the **CTRL** key or the **SHIFT** key to select more than one product.
4. Select **Install product key** in the **Selected Items** menu in the right-side pane to display the **Install Product Key** dialog box.
5. The **Select Product Key** dialog box displays the keys that are available to be installed. Under **Recommended MAKs**, VAMT might display one or more recommended MAKs based on the selected products. If you're installing a MAK, you can select a recommended product key or any other MAK from the **All Product Keys List**. If you aren't installing a MAK, select a product key from the **All Product Keys** list. Use the scroll bar if you want to view the **Description** for each key. When you've selected the product key that you want to install, select **Install Key**. Only one key can be installed at a time.
6. VAMT displays the **Installing product key** dialog box while it attempts to install the product key for the selected products. When the process is finished, the status appears in the **Action Status** column of the dialog box. Select **Close** to close the dialog box. You can also select the **Automatically close when done** check box when the dialog box appears.

   The same status appears under the **Status of Last Action** column in the product list view in the center pane.

> **Note**
>
> Product key installation will fail if VAMT finds mismatched key types or editions. VAMT will display the failure status and will continue the installation for the next product in the list. For more information on choosing the correct product key, see How to Choose the Right Volume License Key for Windows.

Step 9: Activate the client products

1. Select the individual products that you want to activate in the list-view pane.
2. On the menu bar, select **Action**, point to **Activate** and point to **Online activate**. You can also right-click the selected computers(s) to display the **Action** menu, point to **Activate** and point to **Online activate**. You can also select **Activate** in the **Selected Items** menu in the right-hand pane to access the **Activate** option.
3. If you're activating product keys using your current credential, select **Current credential** and continue to step 5. If you're activating products that require an administrator credential that is different from the one you're currently using, select the **Alternate credential** option.
4. Enter your alternate user name and password and select **OK**.
5. The **Activate** option contacts the Microsoft product-activation server over the Internet and requests activation for the selected products. VAMT displays the **Activating products** dialog box until the requested actions are completed.

> **Note**
>
> Installing a MAK and overwriting the GVLK on client products must be done with care. If the RTM version of Windows Vista has been installed on the computer for more than 30 days, then its initial grace period has expired. As a result, it will enter Reduced Functionality Mode (RFM) if online activation is not completed successfully before the next logon attempt. However, you can use online activation to recover properly configured computers from RFM, as long as the computers are available on the network. RFM only applies to the RTM version of Windows Vista or the retail editions of Microsoft Office 2010. Windows Vista with SP1 or later, Windows 7, Windows 8, Windows 10, Windows Server 2008, Windows Server 2008 R2, Windows Server 2012, and volume editions of Office 2010 will not enter RFM.

Related articles

Run online activation

Feedback

Was this page helpful? Yes No

Provide product feedback

Scenario 2: proxy activation

Article • 02/28/2023

In this scenario, the Volume Activation Management Tool (VAMT) is used to activate products that are installed on workgroup computers in an isolated lab environment. For workgroups that are isolated from the larger network, you can perform proxy activation of Multiple Activation Keys (MAKs), KMS Host keys (CSVLKs), Generic Volume License Keys (GVLKs) (or KMS client keys), or retail keys. Proxy activation is performed by installing a second instance of VAMT on a computer in the isolated workgroup. You can then use removable media to transfer VAMT Computer Information Lists (CILXs) between the instance of VAMT in the isolated workgroup and another VAMT host that has internet access. The following diagram shows a Multiple Activation Key (MAK) proxy activation scenario:

Step 1: Install VAMT on a workgroup computer in the isolated lab

1. Install VAMT on a host computer in the isolated lab workgroup. This computer can be running Windows 7, Windows 8, Windows 10, Windows Server 2008 R2, or Windows Server® 2012.

2. Select the VAMT icon in the **Start** menu to open VAMT.

Step 2: Configure the Windows Management Instrumentation firewall exception on target

computers

- Ensure that the Windows Management Instrumentation (WMI) firewall exception has been enabled for all target computers. For more information, see Configure client computers.

> **Note**
>
> To retrieve the license status on the selected computers, VAMT must have administrative permissions on the remote computers and WMI must be accessible through the Windows Firewall. In addition, for workgroup computers, a registry key must be created to enable remote administrative actions under User Account Control (UAC). For more information, see Configure client computers.

Step 3: Connect to a VAMT database

1. If the host computer in the isolated lab workgroup isn't already connected to the database, the **Database Connection Settings** dialog box appears when you open VAMT. Select the server and database that contains the computers in the workgroup.

2. Select **Connect**.

3. If you're already connected to a database, in the center pane VAMT displays an inventory of the products and product keys, and a license overview of the computers in the database. If you need to connect to a different database, select **Successfully connected to the Server** to open the **Database Connection Settings** dialog box. For more information about how to create VAMT databases and adding VAMT data, see Import and export Volume Activation Management Tool data.

Step 4: Discover products

1. In the left-side pane, in the **Products** node, select the product that you want to activate.

2. To open the **Discover Products** dialog box, select **Discover products** in the right-side pane.

3. In the **Discover Products** dialog box, select **Search for computers in the Active Directory** to display the search options, and then select the search options that you want to use. You can search for computers in an Active Directory domain, by individual computer name or IP address, in a workgroup, or by a general LDAP query:

    - To search for computers in an Active Directory domain, select **Search for computers in the Active Directory**. Then under **Domain Filter Criteria**, in the list of domain names, select the name of the domain that you want to search. You can narrow the search further by typing a name in the **Filter by computer name** field to search for specific computers in the domain. This filter supports the asterisk (`*`) wildcard. For example, typing `a*` will display only computer names that start with the letter "a".

    - To search by individual computer name or IP address, select **Manually enter name or IP address**. Then enter the full name or IP address in the **One or more computer names or IP addresses separated by commas** text box. Separate multiple entries with a comma. Both IPv4 and IPv6addressing are supported.

    - To search for computers in a workgroup, select **Search for computers in the workgroup**. Then under **Workgroup Filter Criteria**, in the list of workgroup names, select the name of the workgroup that you want to search. You can narrow the search further by typing a name in the **Filter by computer name** field to search for a specific computer in the workgroup. This filter supports the asterisk (`*`) wildcard. For example, typing `a*` will display only those computer names that start with the letter **a**.

    - To search for computers by using a general LDAP query, select **Search with LDAP query** and enter your query in the text box that appears. VAMT will validate the LDAP query syntax, but will otherwise run the query without extra checks.

4. Select **Search**.

    The **Finding Computers** window appears and displays the search progress as the computers are located.

When the search is complete, the products that VAMT discovers appear in the list view in the center pane.

Step 5: Sort and filter the list of computers

You can sort the list of products so that it's easier to find the computers that require product keys to be activated:

1. On the menu bar at the top of the center pane, select **Group by**, and then select **Product**, **Product Key Type**, or **License Status**.
2. To sort the list further, you can select one of the column headings to sort by that column.
3. You can also use the **Filter** function to narrow your search for computers by clicking **Filter** in the right-side pane to open the **Filter Products** dialog box.
4. In the **Filter Products** dialog box, you can filter the list by computer name, product name, product key type, license status, or by any combination of these options.
    - To filter the list by computer name, enter a name in the **Computer Name** box.
    - To filter the list by product name, product key type, or license status, select the list you want to use for the filter and select an option. If necessary, select **clear all filters** to create a new filter.
5. Select **Filter**. VAMT displays the filtered list in the product list view in the center pane.

Step 6: Collect status information from the computers in the Isolated lab

To collect the status from select computers in the database, you can select computers in the product list view by using one of the following methods:

- To select a block of consecutively listed computers, select the first computer that you want to select, and then select the last computer while pressing the **Shift** key.
- To select computers that aren't listed consecutively, hold down the **Ctrl** key and select each computer for which you want to collect the status information. **To collect status information from the selected computers**
- In the right-side **Actions** pane, select **Update license status** in the **Selected Items** menu and then select a credential option. Choose **Alternate Credentials** only if you're updating products that require administrator credentials that are different from the ones that you used to sign in to the computer. Otherwise, select **Current Credentials** and continue to step 2.If you're supplying alternate credentials, in the

Windows Security dialog box type the appropriate user name and password and then select **OK**.

- VAMT displays the **Collecting product information** dialog box while it collects the license status of all supported products on the selected computers. When the process is finished, the updated license status of each product will appear in the product list view in the center pane.

> ⓘ **Note**
>
> If a computer has more than one supported product installed, VAMT adds an entry for each product. The entry appears under the appropriate product heading.

Step 7: Add product keys

1. Select the **Product Keys** node in the left-side pane, and then select **Add Product Keys** in the right-side pane to open the **Add Product Keys** dialog box.

2. In the **Add Product Keys** dialog box, you can select from one of the following methods to add product keys:

   - To add a single product key, select **Enter product key(s) separated by line breaks**, enter one or more product keys, and then select **Add key(s)**.

   - To import a Comma Separated Values File (CSV) that contains a list of product keys, select **Select a product key to import**, browse to the file location, select **Open** to import the file, and then select **Add Key(s)**.

   The keys that you have added appear in the **Product Keys** list view in the center pane.

Step 8: Install the product keys on the isolated lab computers

1. In the left-side pane, in the **Products** node select the product that you want to install keys onto.

2. If necessary, sort and filter the list of products so that it's easier to find the computers that must have a product key installed. See Step 5: Sort and filter the list of computers.

3. In the **Products** list view pane, select the individual products that must have keys installed. You can use the **CTRL** key or the **SHIFT** key to select more than one product.

4. Select **Install product key** in the **Selected Items** menu in the right-side pane to display the **Install Product Key** dialog box.

5. The **Select Product Key** dialog box displays the keys that are available to be installed. Under **Recommended MAKs**, VAMT might display one or more recommended MAKs based on the selected products. If you're installing an MAK, you can select a recommended product key or any other MAK from the **All Product Keys List**. If you aren't installing a MAK, select a product key from the **All Product Keys** list. Use the scroll bar if you need to view the **Description** for each key. When you've selected the product key that you want to install, select **Install Key**. Only one key can be installed at a time.

6. VAMT displays the **Installing product key** dialog box while it attempts to install the product key for the selected products. When the process is finished, the status appears in the **Action Status** column of the dialog box. Select **Close** to close the dialog box. You can also select the **Automatically close when done** check box when the dialog box appears.

   The same status appears under the **Status of Last Action** column in the product list view in the center pane.

> **Note**
>
> Product key installation will fail if VAMT finds mismatched key types or editions. VAMT displays the failure status and continues the installation for the next product in the list. For more information on choosing the correct product key, see How to Choose the Right Volume License Key for Windows.

> **Note**
>
> Installing a MAK and overwriting the GVLK on client products must be done with care. If the RTM version of Windows Vista has been installed on the computer for more than 30 days, then its initial grace period has expired. As a result, it will enter Reduced Functionality Mode (RFM) if online activation is not completed successfully before the next logon attempt. However, you can use online activation to recover properly configured computers from RFM, as long as the computers are available on the network. RFM only applies to the RTM version of Windows Vista or the retail editions of Microsoft Office 2010.

> Windows Vista with SP1 or later, Windows 7, Windows 8, Windows 10, Windows Server 2008, Windows Server 2008 R2, and Windows Server 2012, and volume editions of Office 2010 will not enter RFM.

Step 9: Export VAMT data to a `.cilx` file

In this step, you export VAMT from the workgroup's host computer and save it in a `.cilx` file. Then you copy the `.cilx` file to removable media so that you can take it to a VAMT host computer that is connected to the internet. In MAK proxy activation, it's critical to retain this file, because VAMT uses it to apply the Confirmation IDs (CIDs) to the proper products.

1. Select the individual products that successfully received a product key in Step 8. If needed, sort and filter the list to find the products.

2. In the right-side **Actions** pane, select **Export list** to open the **Export List** dialog box.

3. In the **Export List** dialog box, select **Browse** to navigate to the `.cilx` file, or enter the name of the `.cilx` file to which you want to export the data.

4. Under **Export options**, select one of the following data-type options:

   - Export products and product keys.

   - Export products only.

   - Export proxy activation data only. Selecting this option ensures that the export contains only the license information required for the proxy web service to obtain CIDs from Microsoft. No Personally Identifiable Information (PII) is contained in the exported `.cilx` file when this selection is selected. This option should be used when an enterprise's security policy states that no information that could identify a specific computer or user may be transferred out of the isolated lab and, therefore, this type of data must be excluded from the `.cilx` file that is transferred to the Core Network VAMT host.

5. If you have selected products to export, and not the entire set of data from the database, select the **Export selected product rows only** check box.

6. Select **Save**. VAMT displays a progress message while the data is being exported. Select **OK** when a message appears and confirms that the export has completed

successfully.

7. If you exported the list to a file on the host computer's hard drive, copy the file to removable media, such as a disk drive, CD/DVD, or USB storage device.

> **Important**
>
> Choosing the **Export proxy activation data only** option excludes Personally Identifiable Information (PII) from being saved in the `.cilx` file. Therefore, the `.cilx` file must be re-imported into the SQL Server database on the isolated lab workgroup's VAMT host computer, so that the CIDs that are requested from Microsoft (discussed in Step 10) can be correctly assigned to the computers in the isolated lab group.

Step 10: Acquire confirmation IDs from Microsoft on the internet connected host computer

1. Insert the removable media into the VAMT host that has internet access.
2. Open VAMT. Make sure you are on the root node, and that the **Volume Activation Management Tool** view is displayed in the center pane.
3. In the right-side **Actions** pane, select **Acquire confirmation IDs for CILX** to open the **Acquire confirmation IDs for file** dialog box.
4. In the **Acquire confirmation IDs for file** dialog box, browse to the location of the `.cilx` file that you exported from the isolated lab host computer, select the file, and then select **Open**. VAMT displays an **Acquiring Confirmation IDs** message while it contacts Microsoft and collects the CIDs.
5. When the CID collection process is complete, VAMT displays a **Volume Activation Management Tool** message that shows the number of confirmation IDs that were successfully acquired, and the name of the file where the IDs were saved. Select **OK** to close the message.

Step 11: Import the `.cilx` file onto the VAMT host within the isolated lab workgroup

1. Remove the storage device that contains the `.cilx` file from the internet-connected VAMT host computer and insert it into the VAMT host computer in the isolated lab.

2. Open VAMT and verify that you're connected to the database that contains the computer with the product keys that you're activating.

3. In the right-side **Actions** pane, select **Import list** to open the **Import List** dialog box.

4. In the **Import list** dialog box, browse to the location of the `.cilx` file that contains the CIDs, select the file, and then select **Open**.

5. Select **OK** to import the file and to overwrite any conflicting data in the database with data from the file.

6. VAMT displays a progress message while the data is being imported. Select **OK** when a message appears and confirms that the data has been successfully imported.

Step 12: Apply the CIDs and activate the isolated lab computers

1. Select the products to which you want to apply CIDs. If needed, sort and filter the list to find the products.

2. In the right-side **Selected Items** menu, select **Activate**, select **Apply Confirmation ID**, and then select the appropriate credential option. If you select the **Alternate Credentials** option, you'll be prompted to enter an alternate user name and password.

   VAMT displays the **Applying Confirmation Id** dialog box while it installs the CIDs on the selected products. When VAMT finishes installing the CIDs, the status appears in the **Action Status** column of the dialog box. Select **Close** to close the dialog box. You can also select the **Automatically close when done** check box when the dialog box appears. The same status appears under the **Status of Last Action** column in the product list view in the center pane.

Step 13: (Optional) Reactivating reimaged computers in the isolated lab

If you have captured new images of the computers in the isolated lab, but the underlying hardware of those computers hasn't changed, VAMT can reactivate those computers using the CIDs that are stored in the database.

1. Redeploy products to each computer, using the same computer names as before.
2. Open VAMT.
3. In the right-side **Selected Items** menu, select **Activate**, select **Apply Confirmation ID**, and then select the appropriate credential option. If you select the **Alternate Credentials** option, you'll be prompted to enter an alternate user name and password.

   VAMT displays the **Applying Confirmation Id** dialog box while it installs the CIDs on the selected products. When VAMT finishes installing the CIDs, the status appears in the **Action Status** column of the dialog box. Select **Close** to close the dialog box. You can also select the **Automatically close when done** check box when the dialog box appears. The same status appears under the **Status of Last Action** column in the product list view in the center pane.

   > **Note**
   >
   > Installing a MAK and overwriting the GVLK on the client products must be done with care. If the Windows activation initial grace period has expired, Windows will enter Reduced Functionality Mode (RFM) if online activation is not completed successfully before the next logon attempt. However, you can use online activation to recover properly configured computers from RFM, as long as the computers are accessible on the network.

   RFM only applies to the RTM version of Windows Vista or the retail editions of Microsoft Office 2010. Windows Vista with SP1 or later, Windows 7, Windows 8, Windows 10, Windows Server 2008, Windows Server 2008 R2, and Windows Server 2012, and volume editions of Office 2010 won't enter RFM.

   > **Note**
   >
   > Reapplying the same CID conserves the remaining activations on the MAK.

Related articles

Run proxy activation

Feedback

Was this page helpful? Yes No

Provide product feedback

Scenario 3: KMS client activation

Article • 07/25/2024

In this scenario, you use the Volume Activation Management Tool (VAMT) to activate Key Management Service (KMS) client keys or Generic Volume License Keys (GVLKs). This type of activation can be performed on either Core Network or Isolated Lab computers. By default, volume license editions of Windows Vista, Windows® 7, Windows 8, Windows 10, Windows Server 2008, Windows Server 2008 R2, Windows Server® 2012, and Microsoft® Office 2010 use KMS for activation. GVLKs are already installed in volume license editions of these products. You don't have to enter a key to activate a product as a GVLK, unless you're converting a MAK-activated product to a KMS activation. For more information, see Install a KMS client key.

The procedure that is described below assumes the following configuration:

- The KMS Service is enabled and available to all KMS clients.
- VAMT has been installed and computers have been added to the VAMT database. See Parts 1 through 4 in either Scenario 1: Online Activation or Scenario 2: Proxy Activation for more information.

Activate KMS clients

1. Open VAMT.
2. To set the KMS activation options, on the menu bar select **View**. Then select **Preferences** to open the **Volume Activation Management Tool Preferences** dialog box.
3. In the **Volume Activation Management Tool Preferences** dialog box, under **KMS Management Services host selection** select from the following options:
    - **Find a KMS host automatically using DNS**. This setting is the default setting. VAMT will instruct the computer to query the Domain Name Service (DNS) to locate a KMS host and perform activation. If the client contains a registry key with a valid KMS host, that value will be used instead.
    - **Find a KMS host using DNS in this domain for supported products**. Select this option if you use a specific domain, and enter the name of the domain.
    - **Use specific KMS host**. Select this option for environments that don't use DNS for KMS host identification, and manually enter the KMS host name and

select the KMS host port. VAMT will set the specified KMS host name and KMS host port on the target computer, and then instruct the computer to perform activation with the specific KMS host.

4. In the left-side pane, in the **Products** node, select the product that you want to activate.

5. In the products list view in the center pane, sort the list if necessary. You can use the **Filter** function to narrow your search for computers by clicking **Filter** in the right-side pane to open the **Filter Products** dialog box.

6. In the **Filter Products** dialog box, you can filter the list by computer name, product name, product key type, license status, or by any combination of these options.

   - To filter the list by computer name, enter a name in the **Computer Name** box.

   - To filter the list by Product Name, Product Key Type, or License Status, select the list you want to use for the filter and select an option. If necessary, select **clear all filters** to create a new filter.

7. Select **Filter**. VAMT displays the filtered list in the center pane.

8. Select the products that you want to activate.

9. Select **Activate** in the **Selected Items** menu in the right-side **Actions** pane, select **Activate**, point to **Volume activate**, and then select the appropriate credential option. If you select the **Alternate Credentials** option, you'll be prompted to enter an alternate user name and password.

10. VAMT displays the **Activating products** dialog box until it completes the requested action. When activation is complete, the status appears in the **Action Status** column of the dialog box. Select **Close** to close the dialog box. You can also select the **Automatically close when done** check box when the dialog box appears.

The same status is shown under the **Status of Last Action** column in the products list view in the center pane.

Related articles

Run KMS activation

Activate using Key Management Service

Install a KMS client key

Feedback

Was this page helpful? Yes No

Provide product feedback

VAMT known issues

Article • 02/28/2023

The current known issues with the Volume Activation Management Tool (VAMT), versions 3.0. and 3.1, include:

- VAMT Windows Management Infrastructure (WMI) remote operations might take longer to execute if the target computer is in a sleep or standby state.
- When you open a Computer Information List (CIL) file that was saved by using a previous version of VAMT, the edition information isn't shown for each product in the center pane. You must update the product status again to obtain the edition information.
- The remaining activation count can only be retrieved for Multiple Activation Key (MAKs).

Workarounds for adding CSVLKs for Windows 10 activation to VAMT 3.1

Another known issue is that when you try to add a Windows 10 Key Management Service (KMS) Host key (CSVLK) or a Windows Server 2012 R2 for Windows 10 CSVLK into VAMT 3.1 (version 10.0.10240.0), you receive the error message shown here.

This issue occurs because VAMT 3.1 doesn't contain the correct Pkconfig files to recognize this kind of key. To work around this issue, use one of the following methods.

Method 1

Don't add the CSVLK to the VAMT 3.1 tool. Instead, use the `slmgr.vbs /ipk <CSVLK>` command to install a CSVLK on a KMS host. In this command, <*CSVLK*> represents the specific key that you want to install. For more information about how to use the `Slmgr.vbs` tool, see Slmgr.vbs options for obtaining volume activation information.

Method 2

On the KMS host computer, perform the following steps:

1. Download the hotfix from July 2016 update rollup for Windows 8.1 and Windows Server 2012 R2.

2. In Windows Explorer, right-click **485392_intl_x64_zip** and extract the hotfix to C:\KB3058168.

3. To extract the contents of the update, run the following command:

```
Windows Command Prompt

expand c:\KB3058168\Windows8.1-KB3058168-x64.msu -f:* C:\KB3058168\
```

4. To extract the contents of Windows8.1-KB3058168-x64.cab, run the following command:

```
Windows Command Prompt

expand c:\KB3058168\Windows8.1-KB3058168-x64.cab -f:pkeyconfig-
csvlk.xrm-ms c:\KB3058168
```

5. In the

   `C:\KB3058168\x86_microsoft-windows-s..nent-sku-csvlk-pack_31bf3856ad364e35_6.3.9600.17815_none_bd26b4f34d049716`

   folder, copy the

   `pkeyconfig-csvlk.xrm-ms`

   file. Paste this file into the

   `C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\VAMT3\pkconfig`

   folder.

6. Restart VAMT.

Feedback

Was this page helpful? Yes No

Provide product feedback

Windows Update for Business reports overview

Article • 03/31/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

Windows Update for Business reports is a cloud-based solution that provides information about your Microsoft Entra joined devices' compliance with Windows updates. Windows Update for Business reports is offered through the Azure portal, and it's included as part of the Windows 10 or Windows 11 prerequisite licenses. Windows Update for Business reports helps you:

- Monitor security, quality, driver, and feature updates for Windows 11 and Windows 10 devices
- Report on devices with update compliance issues
- Analyze and display your data in multiple ways

Benefits of Windows Update for Business reports

Some of the benefits of Windows Update for Business reports are:

- Compatibility with feature updates and Expedite Windows quality updates policies in Intune.
- A new **Alerts** data type to assist you with identifying devices that encounter issues during the update process. Error code information is provided to help troubleshoot update issues.

Currently, Windows Update for Business reports contains the following features:

- Windows Update for Business reports workbook
- Compliance status charts in the Microsoft 365 admin
- Access to the following Windows Update for Business reports tables:
  - UCClient
  - UCClientReadinessStatus
  - UCClientUpdateStatus
  - UCDeviceAlert
  - UCDOAggregatedStatus
  - UCDOStatus
  - UCServiceUpdateStatus

- UCUpdateAlert

- Client data collection to populate the Windows Update for Business reports tables

Limitations

Windows Update for Business reports is a Windows service hosted in Azure that uses Windows diagnostic data. You should be aware that Windows Update for Business reports doesn't meet US Government community compliance (GCC) requirements. For a list of GCC offerings for Microsoft products and services, see the Microsoft Trust Center. Windows Update for Business reports is available in the Azure Commercial cloud, but not available for GCC High or United States Department of Defense customers.

How Windows Update for Business reports works

You'll set up Windows Update for Business reports by enrolling into the service from the Azure portal. Then you'll configure your Microsoft Entra joined devices to send Windows client diagnostic data to the service. Windows Update for Business reports uses Log Analytics in Azure Monitor to store the diagnostic data the clients send. You can use this data for reporting on updates for your devices. Windows Update for Business reports collects system data such as:

- Update deployment progress
- Delivery Optimization usage data
- Windows Update client policy configuration data

The Azure Log Analytics ingestion and retention charges aren't incurred on your Azure subscription for Windows Update for Business reports data. You also choose an Azure Log Analytics workspaces that you own for your client diagnostic data. The collected diagnostic data populates the Windows Update for Business reports tables so you can easily query your data.

Use your Windows Update for Business reports data

Since the data from your clients is stored in a Log Analytics workspace, you can go beyond the standard reports to analyze and display your data in multiple ways. Some of the ways you could display your data include:

- Using the data in custom workbooks that you create
- Building custom Kusto (KQL) queries
- Developing your own custom views by integrating the Log Analytics data into other tools such as:
  - Operations Management Suite
  - Power BI
  - Other tools for querying the data

Next steps

- Review the Windows Update for Business reports prerequisites

---

Feedback

Was this page helpful? Yes No

Provide product feedback

Windows Update for Business reports prerequisites

Article • 06/04/2024 • Applies to: ✅ Windows 11, ✅ Windows 10

Before you begin the process of adding Windows Update for Business reports to your Azure subscription, ensure you meet the prerequisites.

Azure and Microsoft Entra ID

- An Azure subscription with Microsoft Entra ID.
- Devices must be Microsoft Entra joined and meet the below OS, diagnostic, and endpoint access requirements.
  - Devices can be Microsoft Entra joined or Microsoft Entra hybrid joined.
- Devices that are Microsoft Entra registered only (workplace joined) aren't supported with Windows Update for Business reports.
- The Log Analytics workspace must be in a supported region.
- Data in the **Driver update** tab of the workbook is only available for devices that receive driver and firmware updates from Windows Autopatch.

Permissions

Accessing Windows Update for Business reports typcially requires permissions from multiple sources including:

- Microsoft Entra ID or Intune: Used for managing Windows Update for Business services through Microsoft Graph API, such as enrolling into reports
- Azure: Used for controlling access to Azure resources through Azure Resource Management, such as access to the Log Analytics workspace
- Microsoft 365 admin center: Manages access to the Microsoft 365 admin center, which allows only users with certain Microsoft Entra roles access to sign in

Roles that can enroll into Windows Update for Business reports

To enroll into Windows Update for Business reports from the Azure portal or the Microsoft 365 admin center requires one of the following roles:

- Intune Administrator Microsoft Entra role
- Windows Update deployment administrator Microsoft Entra role
- Policy and profile manager Microsoft Intune role

- Microsoft Intune RBAC roles don't allow access to the Microsoft 365 admin center

Azure roles that allow access to the Log Analytics workspace

The data for Windows Update for Business reports is routed to a Log Analytics workspace for querying and analysis. To display or query any of Windows Update for Business reports data, users must have the following roles, or the equivalent permissions for the workspace:

- Log Analytics Reader role can be used to read data
- Log Analytics Contributor role can be used if creating a new workspace or write access is needed

Examples of commonly assigned roles for Windows Update for Business reports users:

Expand table

Roles	Enroll though the workbook	Enroll through Microsoft 365 admin center	Display the workbook	Microsoft 365 admin center access	Create Log Analytics workspace
Intune Administrator + Log Analytics Contributor	Yes	Yes	Yes	Yes	Yes
Windows Update deployment administrator + Log Analytics reader	Yes	Yes	Yes	Yes	No
Policy and profile manager (Intune role)+ Log Analytics reader	Yes	No	Yes	No	No
Log Analytics reader	No	No	Yes	No	No
Global reader + Log Analytics reader	No	No	Yes	Yes	No

> **Note**
>
> The Microsoft Entra roles discussed in this article for the Microsoft 365 admin center access apply specifically to the **Windows** tab of the **Software Updates** page. For more information about the **Microsoft 365 Apps** tab, see Microsoft 365 Apps updates in the admin center.

Operating systems and editions

- Windows 11 Professional, Education, Enterprise, and Enterprise multi-session editions
- Windows 10 Professional, Education, Enterprise, and Enterprise multi-session editions

Windows Update for Business reports only provides data for the standard desktop Windows client version and isn't currently compatible with Windows Server, Surface Hub, IoT, or other versions.

> **Important**
>
> Currently there is a known issue where Windows Update for Business reports doesn't display data for Enterprise multi-session edition devices.

Windows client servicing channels

Windows Update for Business reports supports Windows client devices on the following channels:

- General Availability Channel
- Windows Update for Business reports *counts* Windows Insider Preview devices, but doesn't currently provide detailed deployment insights for them.

Windows operating system updates for client devices

Installing the February 2023 cumulative update, or a later equivalent update, is required for clients to enroll into Windows Update for Business reports. This update helped

enable changes to Windows diagnostic data collection, which Windows Update for Business reports relies on.

For more information about available updates, see Windows 11 release information and Windows 10 release information.

Diagnostic data requirements

At minimum, Windows Update for Business reports requires devices to send diagnostic data at the *Required* level (previously *Basic*). For more information about what data each diagnostic level includes, see Configure Windows diagnostic data in your organization.

The following levels are recommended, but not required:

- The *Enhanced* level for Windows 10 devices.
- The *Optional* level for Windows 11 devices (previously *Full*).

Device names don't appear in Windows Update for Business reports unless you individually opt in devices by using a policy. The configuration script does this action for you, but when using other client configuration methods, set one of the following policies to display device names:

- CSP: System/AllowDeviceNameInDiagnosticData
- Group Policy: **Allow device name to be sent in Windows diagnostic data** under **Computer Configuration\Administrative Templates\Windows Components\Data Collection and Preview Builds**

> **Tip**
>
> Windows Update for Business reports uses **services configuration**, also called OneSettings. Disabling the services configuration can cause some of the client data to be incorrect or missing in reports. For more information, see the **DisableOneSettingsDownloads** policy settings.

Microsoft is committed to providing you with effective controls over your data and ongoing transparency into our data handling practices. For more information about data handling and privacy for Windows diagnostic data, see Configure Windows diagnostic data in your organization and Changes to Windows diagnostic data collection.

Endpoints

Devices must be able to contact the following endpoints in order to authenticate and send diagnostic data:

Expand table

Endpoint	Function
`*v10c.events.data.microsoft.com` `eu-v10c.events.data.microsoft.com` for tenants with billing address in the EU Data Boundary	Connected User Experience and Diagnostic component endpoint for Windows 10, version 1803 and later. DeviceCensus.exe must run on a regular cadence and contact this endpoint in order to receive most information for Windows Update for Business reports.
`umwatsonc.events.data.microsoft.com` `eu-watsonc.events.data.microsoft.com` for tenants with billing address in the EU Data Boundary	Windows Error Reporting (WER), used to provide more advanced error reporting if certain Feature Update deployment failures occur.
`v10.vortex-win.data.microsoft.com`	Connected User Experience and Diagnostic component endpoint for Windows 10, version 1709 or earlier.
`settings-win.data.microsoft.com`	Used by Windows components and applications to dynamically update their configuration. Required for Windows Update functionality.
`adl.windows.com`	Required for Windows Update functionality.
`oca.telemetry.microsoft.com`	Online Crash Analysis, used to provide device-specific recommendations and detailed errors if there are certain crashes.
`login.live.com`	This endpoint facilitates your Microsoft account access and is required to create the primary identifier we use for devices. Without this service, devices aren't visible in the solution. The Microsoft Account Sign-in Assistant service must also be running (wlidsvc).

`ceuswatcab01.blob.core.windows.net`
`ceuswatcab02.blob.core.windows.net`
`eaus2watcab01.blob.core.windows.net`
`eaus2watcab02.blob.core.windows.net`
`weus2watcab01.blob.core.windows.net`
`weus2watcab02.blob.core.windows.net`
	Azure blob data storage.

Note

Enrolling into Windows Update for Business reports from the Azure CLI or enrolling programmatically another way currently isn't supported. You must manually add Windows Update for Business reports to your Azure subscription.

Log Analytics regions

Windows Update for Business reports can use a Log Analytics workspace in the following regions:

Expand table

Compatible Log Analytics regions
Australia Central
Australia East
Australia Southeast
Brazil South
Canada Central
Central India
Central US
East Asia
East US
East US 2
Eastus2euap(canary)
France Central
Japan East
Korea Central
North Central US
North Europe
South Africa North
South Central US

Compatible Log Analytics regions
Southeast Asia
Switzerland North
Switzerland West
UK West
UK south
West Central US
West Europe
West US
West US 2

Next steps

- Enable the Windows Update for Business reports solution in the Azure portal

Feedback

Was this page helpful? Yes No

Provide product feedback

Enable Windows Update for Business reports

Article • 07/09/2024 • Applies to: ✅ Windows 11, ✅ Windows 10

After verifying the prerequisites are met, you can start to set up Windows Update for Business reports. The two main steps for setting up Windows Update for Business reports are:

1. Add Windows Update for Business reports to your Azure subscription. This step has the following phases:
   a. Select or create a new Log Analytics workspace for use with Windows Update for Business reports.
   b. Enroll into Windows Update for Business reports using one of the following methods:
      - Enroll through the Azure Workbook (preferred method)
      - Enroll from the Microsoft 365 admin center.
2. Configure the clients to send data to Windows Update for Business reports. You can configure clients in the following three ways:
   - Use a script
   - Use Microsoft Intune
   - Configure manually

> **ⓘ Important**
>
> Windows Update for Business reports is a Windows service hosted in Azure that uses Windows diagnostic data. You should be aware that Windows Update for Business reports doesn't meet US Government community compliance (GCC) requirements. For a list of GCC offerings for Microsoft products and services, see the Microsoft Trust Center. Windows Update for Business reports is available in the Azure Commercial cloud, but not available for GCC High or United States Department of Defense customers.

Add Windows Update for Business reports to your Azure subscription

Before you configure clients to send data, you need to add Windows Update for Business reports to your Azure subscription so the data can be received. First, you select or create a new Log Analytics workspace to use. Second, you enroll Windows Update for Business reports to the workspace.

Select or create a new Log Analytics workspace for Windows Update for Business reports

Windows Update for Business reports uses an Azure Log Analytics workspaces that you own for storing the client diagnostic data. Identify an existing workspace or create a new one using the following steps:

1. Sign in to the Azure portal at https://portal.azure.com.
    - Although an Azure subscription is required, you won't be charged for ingestion of Windows Update for Business reports data.
2. In the Azure portal, type **Log Analytics** in the search bar. As you begin typing, the list filters based on your input.
3. Select **Log Analytics workspaces**.
4. If you already have a Log Analytics workspace, determine which Log Analytics workspace you'd like to use for Windows Update for Business reports. Ensure the workspace is in a **Compatible Log Analytics region** from the table listed in the prerequisites.
    - Azure Update Management users should use the same workspace for Windows Update for Business reports.
5. If you don't have an existing Log Analytics workspace or you don't want to use a current workspace, create a new workspace in a compatible region.

> **Note**
>
> - You can only map one tenant to one Log Analytics workspace. Mapping one tenant to multiple workspaces isn't supported.
> - If you change the Log Analytics workspace for Windows Update for Business reports, stale data will be displayed for about 24 hours until the new workspace is fully onboarded. You will also need to reconfigure the Windows Update for Business reports settings to enroll again.

Enroll into Windows Update for Business reports

Enroll into Windows Update for Business reports by configuring its settings through either the Azure Workbook or from the Microsoft 365 admin center. Use one of the following methods to enroll into Windows Update for Business reports:

Enroll through the Azure Workbook (recommended method)

1. In the Azure portal, select **Monitor** > **Workbooks** from the menu bar.
    - You can also type **Monitor** in the search bar. As you begin typing, the list filters based on your input.
2. When the gallery opens, select the **Windows Update for Business reports** workbook. If needed, you can filter workbooks by name in the gallery.
3. Select the **Get started** button when prompted by the workbook to open the **Windows Update for Business reports enrollment** flyout.
4. In the flyout, specify which **Subscription** and **Azure Log Analytics Workspace** you want to use for Windows Update for Business reports.
    - If you need to create a new Log Analytics workspace, select **Create new workspace** and follow the prompts to create a new workspace.
5. Select **Save settings** to save the settings and enroll into Windows Update for Business reports.

    > **Tip**
    >
    > If a `403 Forbidden` error occurs, verify the account you're using has permissions to enroll into Windows Update for Business reports.

6. The initial setup can take up to 24 hours. During this time, the workbook will display that it's **Waiting for Windows Update for Business reports data**.
    - Enrolling into Windows Update for Business reports doesn't influence the rate that required data is uploaded from devices. Device connectivity to the internet and how active the device is influences how long it takes before the device appears in reporting. Devices that are active and connected to the internet daily can expect to be fully uploaded within one week (usually less

than 72 hours). Devices that are less active can take up to two weeks before data is fully available.

Enroll through the Microsoft 365 admin center

1. Go to the Microsoft 365 admin center and sign in.
2. Expand **Health**, then select **Software Updates**. You may need to use the **Show all** option to display **Health** in the navigation menu.
   - If you don't see an entry for **Software updates** in the menu, try going to this URL: https://admin.microsoft.com/Adminportal/Home#/softwareupdates .
3. In the **Software Updates** page, select the **Windows** tab.
4. When you select the **Windows** tab for the first time, you'll be asked to **Configure Settings**. This tab is populated by data from Windows Update for Business reports. Verify or supply the following information about the settings for Windows Update for Business reports:
   - The Azure subscription
   - The Log Analytics workspace
5. The initial setup can take up to 24 hours. During this time, the **Windows** tab will display that it's **Waiting for Windows Update for Business reports data**.
   - Enrolling into Windows Update for Business reports doesn't influence the rate that required data is uploaded from devices. Device connectivity to the internet and how active the device is influences how long it will take before the device appears in reporting. Devices that are active and connected to the internet daily can expect to be fully uploaded within one week (usually less than 72 hours). Devices that are less active can take up to two weeks before data is fully available.
6. After the initial setup is complete, the **Windows** tab will display your Windows Update for Business reports data in the charts.

> **Note**
>
> The device counts in the **Windows** tab may vary from the **Microsoft 365 Apps** tab since their requirements are different.

Next steps

Once you've added Windows Update for Business reports to a workspace in your Azure subscription and configured the settings through the Microsoft 365 admin center, you'll need to configure any devices you want to monitor. Enroll devices into Windows Update for Business reports using any of the following methods:

- Configure clients with a script
- Configure clients manually
- Configure clients with Microsoft Intune

Feedback

Was this page helpful? Yes No

Provide product feedback

Configuring devices through the Windows Update for Business reports configuration script

Article • 07/09/2024 • Applies to: ✅ Windows 11, ✅ Windows 10

The Windows Update for Business reports configuration script is the recommended method of configuring devices to send data to Microsoft for use with Windows Update for Business reports. The script configures the registry keys backing policies, ensures required services are running, and more. This script is a recommended complement to configuring the required policies documented in Manually configure devices for Windows Update for Business reports, as it can provide feedback on whether there are any configuration issues outside of policies being configured.

About the script

The configuration script configures registry keys directly. Registry keys can potentially be overwritten by policy settings like Group Policy or MDM. *Reconfiguring devices with the script doesn't reconfigure previously set policies, both in the case of Group Policy and MDM*. If there are conflicts between your Group Policy or MDM configurations and the required configurations listed in Manually configuring devices for Windows Update for Business reports, device data might not appear in Windows Update for Business reports correctly.

You can download the script from the Microsoft Download Center.

How this script is organized

This script's two primary files are `ConfigScript.ps1` and `RunConfig.bat`. You configure `RunConfig.bat` according to the directions in the `.bat` itself, which will then run `ConfigScript.ps1` with the parameters entered to `RunConfig.bat`. There are two ways of using the script: in **Pilot** mode or **Deployment** mode.

> **Important**
>
> PsExec is used to run the script in the system context. Once the device is configured, remove PsExec.exe from the device.

How to use this script

Edit the `RunConfig.bat` file to configure the following variables, then run the edited .bat file:

Expand table

Variable	Allowed values and description	Example
runMode	**Pilot** (default): Verbose mode with additional diagnostics and logging. Pilot mode is best for a testing run of the script or for troubleshooting. **Deployment**: Doesn't run any additional diagnostics or add extra logging	`runMode=Pilot`
logPath	Path where the logs are saved. The default location of the logs is `.\UCLogs`.	`logPath=C:\temp\logs`
logMode	**0**: Log to the console only **1** (default): Log to file and console. **2**: Log to file only.	`logMode=2`
DeviceNameOptIn	**true** (default): Device name is sent to Microsoft. **false**: Device name isn't sent to Microsoft.	`DeviceNameOptIn=true`
ClientProxy	**Direct** (default): No proxy is used. The connection to the endpoints is direct. **System**: The system proxy, without authentication, is used. This type of proxy is typically configured with netsh and can be verified using `netsh winhttp show proxy`. **User**: The proxy is configured through IE and it might or might not require user authentication. For more information, see How the Windows Update client determines which proxy server to use to connect to the Windows Update website	`ClientProxy=Direct`
source	Used by the .bat file and PowerShell script to locate dependencies. It's recommended that you don't change this value.	`source=%~dp0`

Script errors

Expand table

Error	Description
1	Unexpected error
12	CheckVortexConnectivity failed, check the log output for more information.
12	Unexpected failure when running CheckVortexConnectivity.
16	Reboot is pending on device. Restart the device then re rerun the script.
17	Unexpected exception in CheckRebootRequired.
27	Not system account.
30	Unable to disable Enterprise Auth Proxy. This registry value must be 0 for UTC to operate in an authenticated proxy environment.
34	Unexpected exception when attempting to check proxy settings.
35	Unexpected exception when checking user proxy.
37	Unexpected exception when collecting logs.
40	Unexpected exception when checking and setting telemetry.
41	Unable to impersonate logged-on user.
42	Unexpected exception when attempting to impersonate logged-on user.
43	Unexpected exception when attempting to impersonate logged-on user.
44	Error when running CheckDiagTrack service.
45	DiagTrack.dll not found.
50	DiagTrack service not running.
51	Unexpected exception when attempting to run Census.exe.
52	Couldn't find Census.exe.
54	Microsoft Account Sign In Assistant (MSA) Service disabled.
55	Failed to create new registry path for SetDeviceNameOptIn.
56	Failed to create property for SetDeviceNameOptIn at registry path.
57	Failed to update value for SetDeviceNameOptIn.
58	Unexpected exception in SetDeviceNameOptIn.
59	Failed to delete LastPersistedEventTimeOrFirstBoot property at registry path when attempting to clean up OneSettings.

Error	Description
60	Failed to delete registry key when attempting to clean up OneSettings.
61	Unexpected exception when attempting to clean up OneSettings.
62	AllowTelemetry registry key isn't the correct type of REG_DWORD.
63	AllowTelemetry isn't set to the appropriate value and it couldn't be set by the script.
64	AllowTelemetry isn't the correct type of REG_DWORD.
66	Failed to verify UTC connectivity and recent uploads.
67	Unexpected failure when verifying UTC CSP.
99	Device isn't Windows 10 or Windows 11.
100	Device must be Microsoft Entra joined or Microsoft Entra hybrid joined to use Windows Update for Business reports.
101	Check Microsoft Entra join failed with unexpected exception.
102	DisableOneSettingsDownloads policy shouldn't be enabled. Please disable this policy.

Next steps

Use Windows Update for Business reports

Feedback

Was this page helpful? Yes No

Provide product feedback

Manually configure devices for Windows Update for Business reports

Article • 07/09/2024 • Applies to: ✅ Windows 11, ✅ Windows 10

There are many requirements to consider when manually configuring devices for Windows Update for Business reports. These requirements can potentially change with later versions of Windows client. When any configuration requirements change, we'll update the Windows Update for Business reports configuration script. If that happens, you only need to redeploy the script.

The requirements are separated into different categories:

1. Ensuring the **required policies** for Windows Update for Business reports are correctly configured.
2. Devices in every network topography must send data to the **required endpoints** for Windows Update for Business reports. For example, devices in both main and satellite offices, which might have different network configurations, must be able to reach the endpoints.
3. Ensure **Required Windows services** are running or are scheduled to run. For proper functionality, leave Windows services set to their out-of-box default configurations.

Required policies

The Windows Update for Business reports service has several policies that you need to configure appropriately. These policies allow Microsoft to process your devices and show them in Windows Update for Business reports. The policies are listed in the following subsections, separated by mobile device management (MDM) or group policy.

The following definitions apply for both tables:

- **Policy**: The location and name of the policy.
- **Value**: Set the policy to this value. Windows Update for Business reports requires at least *Required* (previously *Basic*) diagnostic data, but can function with *Enhanced* or *Optional* (previously *Full*).
- **Function**: Details for why the policy is required and what function it serves for Windows Update for Business reports. It also details a minimum version the policy requires, if any.

MDM policies

Each MDM policy links to more detailed documentation in the configuration service provider (CSP) hierarchy.

Expand table

Policy	Data type	Value	Function	Required or recommended
System/AllowTelemetry	Integer	`1`: Basic (Required)	Configures the device to send the minimum required diagnostic data.	Required
System/ConfigureTelemetryOptInSettingsUx	Integer	`1`: Disable diagnostic data opt-in settings	Determines whether users of the device can adjust diagnostic data to levels lower than you define by the *AllowTelemetry* policy. Set the recommended value to disable opt-in settings, or users can change the effective diagnostic data level that might not be sufficient.	Recommended
System/AllowDeviceNameInDiagnosticData	Integer	`1`: Allowed	Allows the device to send its name with Windows diagnostic data. If you don't configure this policy or set it to `0`: Disabled, then the data doesn't include the device name. If the data doesn't include the device name, you can't see	Recommended

Policy	Data type	Value	Function	Required or recommended
			the device in Windows Update for Business reports. In this instance, the reports show `#` instead.	
System/ConfigureTelemetryOptInChangeNotification	Integer	`1`: Disabled	Disables user notifications that appear for changes to the diagnostic data level.	Recommended

Group policies

All group policies that you need to configure for Windows Update for Business reports are under the following path: **Computer Configuration** > **Administrative Templates** > **Windows Components** > **Data Collection and Preview Builds**. All of these policies must be in the *Enabled* state and set to the defined *Value*.

Expand table

Policy	Value	Function	Required or recommended
Allow Diagnostic Data	Send required diagnostic data	Configures the device to send the minimum required diagnostic data.	Required
Configure diagnostic data opt-in setting user interface	Disable diagnostic data opt-in settings	Determines whether users of the device can adjust diagnostic data to levels lower than you define by the *Allow Diagnostic Data* policy. Set the recommended value to disable opt-in settings, or users can change the effective diagnostic data level that might not be sufficient.	Recommended
Allow device name to be sent in Windows diagnostic data	Enabled	Allows the device to send its name with Windows diagnostic data. If you don't configure this policy or set it to *Disabled*, then the data doesn't include the device name. If the data doesn't include the device name, you can't see the device in Windows Update for Business reports. In this instance, the reports show `#` instead.	Recommended

Policy	Value	Function	Required or recommended
Configure diagnostic data opt-in change notifications	Disable diagnostic data change notifications	Disables user notifications that appear for changes to the diagnostic data level.	Recommended

Required endpoints

Devices must be able to contact the following endpoints in order to authenticate and send diagnostic data:

Expand table

Endpoint	Function
`*v10c.events.data.microsoft.com` `eu-v10c.events.data.microsoft.com` for tenants with billing address in the EU Data Boundary	Connected User Experience and Diagnostic component endpoint for Windows 10, version 1803 and later. DeviceCensus.exe must run on a regular cadence and contact this endpoint in order to receive most information for Windows Update for Business reports.
`umwatsonc.events.data.microsoft.com` `eu-watsonc.events.data.microsoft.com` for tenants with billing address in the EU Data Boundary	Windows Error Reporting (WER), used to provide more advanced error reporting if certain Feature Update deployment failures occur.
`v10.vortex-win.data.microsoft.com`	Connected User Experience and Diagnostic component endpoint for Windows 10, version 1709 or earlier.
`settings-win.data.microsoft.com`	Used by Windows components and applications to dynamically update their configuration. Required for Windows Update functionality.
`adl.windows.com`	Required for Windows Update functionality.
`oca.telemetry.microsoft.com`	Online Crash Analysis, used to provide device-specific recommendations and detailed errors if there are certain crashes.
`login.live.com`	This endpoint facilitates your Microsoft account access and is required to create the primary identifier we use for devices. Without this service, devices aren't visible in the solution. The Microsoft Account Sign-in Assistant service must also be running (wlidsvc).

`ceuswatcab01.blob.core.windows.net`
`ceuswatcab02.blob.core.windows.net`
`eaus2watcab01.blob.core.windows.net`
	Azure blob data storage.

Endpoint	Function

`eaus2watcab02.blob.core.windows.net`
`weus2watcab01.blob.core.windows.net`
`weus2watcab02.blob.core.windows.net`	

Required services

Many Windows services are required for Windows Update for Business reports to see device data. Allow all default services from the out-of-box experience to remain running. Use the Windows Update for Business reports configuration script to check whether required services are running or are allowed to run automatically.

Next steps

Use Windows Update for Business reports

Feedback

Was this page helpful? Yes No

Provide product feedback

Configuring Microsoft Intune devices for Windows Update for Business reports

Article • 03/09/2023 •

Applies to: ✅ Windows 11 and Windows 10 devices managed by Microsoft Intune

This article is targeted at configuring devices enrolled to Microsoft Intune for Windows Update for Business reports, within Microsoft Intune itself. Configuring devices for Windows Update for Business reports in Microsoft Intune breaks down to the following steps:

1. Create a configuration profile for devices you want to enroll. The configuration profile contains settings for all the Mobile Device Management (MDM) policies that must be configured.
2. Wait for data to populate. The length of this process depends on the computer being on, connected to the internet, and correctly configured. Some data types take longer to appear than others. For more information, see Use Windows Update for Business reports.

> **Tip**
>
> - If you need to troubleshoot client enrollment, consider deploying the configuration script as a Win32 app to a few devices and reviewing the logs it creates. Additional checks are performed with the script to ensure devices are correctly configured.
> - Intune provides compliance reports and they have their own prerequisites for use. The number of devices that appear in the Intune reports may also vary from the Windows Update for Business reports. For more information, see Intune compliance reports for updates.

Create a configuration profile

Create a configuration profile that will set the required policies for Windows Update for Business reports. There are two profile types that can be used to create a configuration profile for Windows Update for Business reports (select one):

- The settings catalog

- Template for a custom OMA URI-based profile

Settings catalog

1. In the Intune admin center, go to **Devices** > **Windows** > **Configuration profiles**.
2. On the **Configuration profiles** view, select **Create profile**.
3. Select the following options, then select **Create** when you're done:
   - **Platform**: Windows 10 and later
   - **Profile type**: Settings Catalog
4. You're now on the Configuration profile creation page. On the **Basics** tab, provide a **Name** and **Description** for the profile.
5. On the **Configuration settings** page, you'll add multiple settings from the **System** category. Using the **Settings picker**, select the **System** category.
6. Add the following required settings and values the **System** category:
   - **Setting**: Allow Telemetry
     - **Value**: Basic
       - Basic is the minimum value, but it can be safely set to a higher value. Basic is also known as required diagnostic data.
7. Add the following recommended settings and values from the **System** category:

   > **Note**
   >
   > These settings aren't required, but they're recommended to ensure that users of the device cannot override the diagnostic data level of the device.

   - **Setting**: Configure Telemetry Opt In Settings Ux
     - **Value**: Disabled
       - By turning this setting on, you're disabling the ability for a user to potentially override the diagnostic data level of devices such that data won't be available for those devices in Windows Update for Business reports.
   - **Setting**: Configure Telemetry Opt In Change Notification
     - **Value**: Disabled
       - By turning this setting on, you're disabling notifications of diagnostic data changes.

- **Setting**: Allow device name to be sent in Windows diagnostic data
  - **Value**: Allowed
    - If this policy is disabled, the device name won't be sent and won't be visible in Windows Update for Business reports.

8. Continue through the next set of tabs **Scope tags**, **Assignments**, and **Applicability Rules** to assign the configuration profile to devices you wish to enroll.

9. Review the settings and then select **Create**.

Custom OMA URI-based profile

1. In the Intune admin center, go to **Devices** > **Windows** > **Configuration profiles**.

2. On the **Configuration profiles** view, select **Create profile**.

3. Select the following options, then select **Create** when you're done:

   - **Platform**:Windows 10 and later
   - **Profile type**: Templates
   - **Template name**: Custom

4. You're now on the Configuration profile creation screen. On the **Basics** tab, provide a **Name** and **Description**.

5. On the **Configuration settings** page, you'll be adding multiple OMA-URI Settings that correspond to the policies described in Manually configuring devices for Windows Update for Business reports.

   Required settings:
   a. Add a setting configuring the **Windows Diagnostic Data level** for devices:

      - **Name**: Allow Telemetry
      - **Description**: Sets the maximum allowed diagnostic data to be sent to Microsoft, required for Windows Update for Business reports.
      - **OMA-URI**: `./Vendor/MSFT/Policy/Config/System/AllowTelemetry`
      - **Data type**: Integer
      - **Value**: 1
        - *1 is the minimum value meaning required or basic diagnostic data, but it can be safely set to a higher value.*

   Recommended settings, but not required:
   a. Add settings for **Disabling devices' Diagnostic Data opt-in settings interface**. If these aren't disabled, users of each device can potentially override the

diagnostic data level of devices such that data won't be available for those devices in Windows Update for Business reports:

- **Name**: Disable Telemetry opt-in interface
- **Description**: Disables the ability for end users of devices can adjust diagnostic data to levels lower than defined by the Allow Telemetry setting.
- **OMA-URI**: `./Vendor/MSFT/Policy/Config/System/ConfigureTelemetryOptInSettingsUx`
- **Data type**: Integer
- **Value**: 1

b. Add a setting to **Allow device name in diagnostic data**; otherwise, the device name won't be in Windows Update for Business reports:

- **Name**: Allow device name in Diagnostic Data
- **Description**: Allows device name in Diagnostic Data.
- **OMA-URI**: `./Vendor/MSFT/Policy/Config/System/AllowDeviceNameInDiagnosticData`
- **Data type**: Integer
- **Value**: 1

c. Add a setting to **Configure Telemetry Opt In Change Notification**. Diagnostic data opt-in change notifications won't appear when changes occur.

- **Name**: Configure Telemetry Opt In Change Notification
- **Description**: Disables Telemetry Opt In Change Notification
- **OMA-URI**: `./Vendor/MSFT/Policy/Config/System/ConfigureTelemetryOptInChangeNotification`
- **Data type**: Integer
- **Value**: 1

6. Continue through the next set of tabs **Scope tags**, **Assignments**, and **Applicability Rules** to assign the configuration profile to devices you wish to enroll.

7. Review the settings and then select **Create**.

Deploy the configuration script

The Windows Update for Business reports Configuration Script is a useful tool for properly enrolling devices in Windows Update for Business reports, though it isn't strictly necessary. It checks to ensure that devices have the required services running

and checks connectivity to the endpoints detailed in the section on Manually configuring devices for Windows Update for Business reports. You can deploy the script as a Win32 app. For more information, see Win32 app management in Microsoft Intune.

> **Note**
>
> Using the script is optional when configuring devices through Intune. The script can be leveraged as a troubleshooting tool to ensure that devices are properly configured for Windows Update for Business reports.

When you deploy the configuration script as a Win32 app, you won't be able to retrieve the results of logs on the device without having access to the device, or saving results of the logs to a shared filesystem. We recommend deploying the script in pilot mode to a subset of devices that you can access. After following this guidance, you can deploy the configuration script in deployment mode as a Win32 app to all Windows Update for Business reports devices.

Next steps

Use Windows Update for Business reports

Feedback

Was this page helpful? Yes No

Provide product feedback

Windows Update for Business reports workbook

Article • 01/29/2024 • Applies to: ✅ Windows 11, ✅ Windows 10

Windows Update for Business reports presents information commonly needed by updates administrators in an easy-to-use format. Windows Update for Business reports uses Azure Workbooks to give you a visual representation of your compliance data. The workbook is broken down into tab sections:

- Summary
- Quality updates
- Feature updates
- Delivery Optimization
- Driver updates

Open the Windows Update for Business reports workbook

To access the Windows Update for Business reports workbook:

1. In the Azure portal, select **Monitor** > **Workbooks** from the menu bar.

- You can also type **Monitor** in the search bar. As you begin typing, the list filters based on your input.

2. When the gallery opens, select the **Windows Update for Business reports** workbook. If needed, you can filter workbooks by name in the gallery.

3. When the workbook opens, you may need to specify which **Subscription** and **Workspace** you used when enabling Windows Update for Business reports.

> **Important**
>
> Don't pin the Windows Update for Business reports workbook to an Azure dashboard. Using a pinned report loads an older copy of the report and it won't display any updates to the report template.

Summary tab

The **Summary** tab gives you a brief high-level overview of the devices that you've enrolled into Windows Update for Business reports. The **Summary** tab contains tiles above the **Overall security update status** chart.

Summary tab tiles

Each of these tiles contains an option to **View details**. When **View details** is selected for a tile, a flyout appears with additional information.

Expand table

Tile name	Description	View details description
Enrolled devices	Total number of devices that are enrolled into Windows Update for Business reports	Displays multiple charts about the operating systems (OS) for enrolled devices: **OS Version** **OS Edition**

Tile name	Description	View details description
		OS Servicing Channel OS Architecture
Active alerts	Total number of active alerts on enrolled devices	Displays the top three active alert subtypes and the count of devices in each. Select the count of **Devices** to display a table of the devices. This table is limited to the first 1000 rows. Select `...` to export the full list, or display the query in Log Analytics. Select an **AlertSubtype** to display a list containing: - Each **Error Code** in the alert subtype - A **Description** of the error code - A **Recommendation** to help you remediate the error code - A count of **Devices** with the specific error code
Windows 11 adoption	Number of devices that are running Windows 11	Displays the following items: - **Windows 11 Device Count** chart, broken down by Windows 11 version - **Windows 11 Eligibility Status** contains a **Readiness status** chart that lists the count of devices by OS version that are either capable or not capable of running Windows 11. - The **Device List** allows you to choose a Windows 11 **Ineligibility Reason** to display devices that don't meet the selected requirement.

Summary tab charts

The charts displayed in the **Summary** tab give you a general idea of the overall status of your devices. The two charts displayed include:

- **Overall security update status**: Gives you general insight into of the current update compliance state of your enrolled devices. For instance, if the chart shows a large number of devices are missing multiple security updates, it may indicate an issue in the software update process.

- **Feature update status**: Gives you a general understanding of how many devices are eligible for feature updates based on the operating system lifecycle.

Quality updates tab

The **Quality updates** tab displays generalized data at the top by using tiles. The quality update data becomes more specific as you navigate lower in this tab. The top of the **Quality updates** tab contains tiles with the following information and drill-down options:

Expand table

Tile name	Description	Drill-in description
Latest security update	Count of devices that have reported successful installation of the latest security update.	- Select **View details** to display a flyout with a chart that displays the first 1000 items. - Select `...` from the flyout to export the full list, or display the query in Log Analytics.
Missing one security update	Count of devices that haven't installed the latest security update.	- Select **View details** to display a flyout with a chart that displays the first 1000 items. - Select `...` from the flyout to export the full list, or display the query in Log Analytics.
Missing multiple security updates	Count of devices that are missing two or more security updates.	- Select **View details** to display a flyout with a chart that displays the first 1000 items. - Select `...` from the flyout to export the full list, or display the query in Log Analytics.
Active alerts	Count of active update and device alerts for quality updates.	

Tile name	Description	Drill-in description
Expedite status	Overview of the progress for the expedited deployments of the latest security update.	Select **View details** to display a flyout with two tabs: **Deployments** and **Readiness** - The **Deployments** tab contains a chart that displays the total progress of each deployment, number of alerts, and count of devices. • Select the count from the **Alerts** column to display the alerts, by name, for the deployment. Selecting the device count for the alert name displays a list of devices with the alert. • Select the count in the **TotalDevices** column to display a list of clients and their information for the deployment. - The **Readiness** tab contains a chart that displays the number of devices that are **Eligible** and **Ineligible** to install expedited updates. The **Readiness** tab also contains a table listing the deployments for expedited updates. • Select the count from the **Alerts** column to display devices with a status of **RegistrationMissingUpdateClient**, which means the device is missing the Update Health Tools. The Update Health Tools are installed starting with KB4023057 or from a stand-alone package from the Microsoft Download Center. Example PowerShell script to verify tools installation: `Get-CimInstance -ClassName Win32_Product | Where-Object {$_.Name -match "Microsoft Update Health Tools"}` • Select the count of **TotalDevices** to display a list of devices in the deployment.

Below the tiles, the **Quality updates** tab is subdivided into **Update status** and **Device status** groups. These different chart groups allow you to easily discover trends in compliance data. For instance, you may remember that about third of your devices were in the installing state yesterday, but this number didn't change as much as you were expecting. That unexpected trend may cause you to investigate and resolve a potential issue before end users are impacted.

Update status group for quality updates

The **Update status** group for quality updates contains the following items:

- **Update states for all security releases**: The update states for the last 3 security updates are used to populate this chart. The total number of update states is approximately 3 times the number of devices that have reported update data to Windows Update for Business reports in the past 30 days.
- **Update alerts for all security releases**: Chart containing the count of active errors and warnings for security updates.

The **Update deployment status** table displays the quality updates for each operating system version that were released within the last 60 days. For each update, drill-in further by selecting a value from the following columns:

Expand table

Column name	Description	Drill-in description
Alerts	Number of different error codes encountered by devices for the update.	Selecting this number lists the alert name for each error code and a count of devices with the error. Select the device count to display a list of devices that have an active alert for the error code.
KB Number	KB number for the update	Selecting the KB number will open the support information webpage for the update.
Total devices	Number of devices that have been offered the update, or are installing, have installed, or canceled the update.	Selecting the device count opens a device list table. This table is limited to the first 1000 rows. Select `...` to export the full list, or display the query in Log Analytics.

Device status group for quality updates

The **Device status** group for quality updates contains the following items:

- **OS build number**: Chart containing a count of devices by OS build that are getting security updates.
- **Device alerts**: Chart containing the count of active device errors and warnings for quality updates.
- **Device compliance status**: Table containing a list of devices getting security updates and update installation information including active alerts for the devices.
  - This table is limited to the first 1000 rows. Select `...` to export the full list, or display the query in Log Analytics.

Feature updates tab

The **Feature updates** tab displays generalized data at the top by using tiles. The feature update data becomes more specific as you navigate lower in this tab. The top of the **Feature updates** tab contains tiles with the following information:

- **In service feature update**: Count of devices that are installed with a supported version of a Windows feature update.
- **End of service feature update**: Count of devices that don't have a supported version of a Windows feature update installed. For more information, see the Windows lifecycle FAQ.
- **Nearing EOS** Count of devices that are within 18 months of their end of service date.
- **Active alerts**: Count of active update and device alerts for feature updates.

Just like the **Quality updates** tab, the **Feature updates** tab is also subdivided into **Update status** and **Device status** groups below the tiles. Selecting **View details** on any of the tiles displays a flyout with a chart that displays the first 1000 items. Select `...` from the flyout to export the full list, or display the query in Log Analytics.

Update status group for feature updates

The **Update status** group for feature updates contains the following items:

- **Target version**: Chart containing count of devices per targeted operating system version.
- **Safeguard holds**: Chart containing count of devices per operating system version that are under a safeguard hold for a feature update
- **Update alerts**: Chart containing the count of active errors and warnings for feature updates.

Update deployment status table for feature updates displays the installation status by targeted operating system version. For each operating system version targeted, the following columns are available:

Expand table

Column name	Description	Drill-in description
Total progress	Percentage of devices that installed the targeted operating system version feature update within the last 30 days.	A bar graph is included in this column. Use the **Total devices** drill-in for additional information.
Alerts	Number of different error codes encountered by devices for the	Selecting this number lists the alert name for each error code and a count of devices

Column name	Description	Drill-in description
	update.	with the error. Select the device count to display a list of devices that have an active alert for the error code.
Total Devices	Count of devices for each targeted operating system version that have been offered the update, or are installing, have installed, or canceled the feature update.	Selecting the device count opens a device list table. This table is limited to the first 1000 rows. Select `...` to export the full list, or display the query in Log Analytics.

Device status group for feature updates

The **Device status** group for feature updates contains the following items:

- **Windows 11 readiness status**: Chart containing how many devices that have a status of capable, not capable, or unknown for Windows 11 readiness.
- **Device alerts**: Count of active device alerts for feature updates in each alert classification.
- **Device compliance status**: Table containing a list of devices getting a feature update and installation information including active alerts for the devices.
  - This table is limited to the first 1000 rows. Select `...` to export the full list, or display the query in Log Analytics.

Driver updates tab

The **Driver update** tab provides information on driver and firmware update deployments from Windows Autopatch. Generalized data is at the top of the page in tiles. The data becomes more specific as you navigate lower in this tab. The top of the driver updates tab contains tiles with the following information:

Devices taking driver updates: Count of devices that are installing driver and firmware updates. **Approved updates**: Count of approved driver updates **Total policies**: The total number of deployment polices for driver and firmware updates from Windows Autopatch **Active alerts**: Count of active alerts for driver deployments

Selecting **View details** on any of the tiles displays a flyout with a chart that displays the first 1000 items. Select `...` from the flyout to export the full list, or display the query in Log Analytics.

Just like the Quality updates and Feature updates tabs, the **Driver updates** tab is also subdivided into **Update status** and **Device status** groups below the tiles. These different chart groups allow you to easily discover trends in compliance data.

Update status group for drivers

The **Update status** group for driver updates contains the following items:

- **Update states for all driver updates**: Chart containing the number of driver updates in a specific state, such as installing.
- **Distribution of Driver Classes**: Chart containing the number of drivers in a specific class.
- **Update alerts for all driver updates**: Chart containing the count of active errors and warnings for driver updates.

The **Update deployment status** table displays information about deployed driver updates for your devices. Drill-in further by selecting a value from the **TotalDevices** column to display the status of a specific driver for a specific policy along with information about the installation status for each device.

Device status group for driver updates

The **Device status** group for driver updates contains the following items:

- **Device alerts**: Count of active device alerts for driver updates in each alert classification.
- **Device compliance status**: Table containing a list of devices getting a driver update and installation information including active alerts for the devices.
  - This table is limited to the first 1000 rows. Select `...` to export the full list, or display the query in Log Analytics.

Delivery Optimization

The **Delivery Optimization** tab provides a summarized view of bandwidth efficiencies. This new revised report also includes Microsoft Connected Cache information. For more information, see Delivery Optimization data in Windows Update for Business reports.

At the top of the report, tiles display the following information:

- Total bandwidth savings percentage
- The percentage of the saved bandwidth broken down by peer-to-peer and Microsoft Connected Cache
- Device counts showing percentages of bytes delivered between peer-to-peer and Connected Cache
- The breakdown of total downloaded GBs.

The Delivery Optimization tab is further divided into the following groups:

- **Device Configuration**: A chart differentiating the number of devices with and without peer-to-peer. And, a table of Download Mode configuration breakdown between numbers of devices. When selected, the devices within that group can be viewed, filtered in a separate table.
- **Content Distribution**: Includes charts showing percentage volumes and GB volumes by source by content types. All content types are linked to a table for deeper filtering by **ContentType**, **AzureADTenantId**, and **GroupID**.
- **Efficiency By Group**: This view provides filters commonly used ways of grouping devices. The provided filters include: **GroupID**, **City**, **Country**, and **ISP**.

Understanding update states

Updates can go though many phases from when they're initially deployed to being installed on the device. Transition from one state to another can be rapid, which makes some states less likely to be displayed in reports. The workbook can report the following high-level states for a device update:

- **Offering**: The update is being offered to the device for installation
- **Installing**: The update is in the process of being installed on the device
- **Installed**: The update has been installed on the device
- **Canceled**: The update was canceled from the deployment service before it was installed
- **Uninstalled**: The update was uninstalled from the device by either an admin or a user
- **OnHold**: The update was put on hold from the deployment service before it was installed
- **Unknown**: This state occurs when there's a record for the device in the UCClient table, but there isn't a record for the specific update for the specific device in the UCClientUpdateStatus table. This means that there is no record of the update for the device in question.

Customize the workbook

Since the Windows Update for Business reports workbook is an Azure Workbook template, it can be customized to suit your needs. If you open a template, make some adjustments, and save it, the template is saved as a workbook. This workbook appears in green. The original template is left untouched. For more information about workbooks, see Get started with Azure Workbooks.

Next steps

- Explore the Windows Update for Business reports schema
- Review Feedback, support, and troubleshooting information for Windows Update for Business reports

---

Feedback

Was this page helpful? Yes No

Provide product feedback

Delivery Optimization data in Windows Update for Business reports

Applies to: ✅ Windows 11, ✅ Windows 10

Delivery Optimization (DO) is a Windows feature that can be used to reduce bandwidth consumption by sharing the work of downloading updates among multiple devices in your environment. You can use DO with many other deployment methods, but it's a cloud-managed solution, and access to the DO cloud services is a requirement.

Windows Update for Business reports provides Delivery Optimization information in the following places:

- The Windows Update for Business reports workbook
- UCDOAggregatedStatus
- UCDOStatus

Windows Update for Business reports doesn't include Delivery Optimization data for Windows Insider devices.

Delivery Optimization terms

Windows Update for Business reports uses the following Delivery Optimization terms:

- **Peer**: A device in the solution
- **Peering 'ON'** - Devices where DO peer-to-peer is enabled in one of the following modes:
  - LAN (1)
  - Group (2)
  - Internet (3)
- **Peering 'OFF'**: Devices where DO peer-to-peer is disabled, set to one of the following modes:
  - HTTP Only (0)
  - Simple Mode (99)
  - Bypass (100), deprecated in Windows 11
- **Bandwidth savings**: The percentage of bandwidth that was downloaded from alternate sources (Peers or Microsoft Connected Cache) out of the total amount of data downloaded.
- If bandwidth savings are <= 60%, a *Warning* icon is displayed

- When bandwidth savings are <10%, an *Error* icon is displayed.
- **Configurations**: Based on the DownloadMode configuration set via MDM, Group Policy, or end-user via the user interface.
- **P2P Device Count**: The device count is the number of devices configured to use peering.
- **Microsoft Connected Cache**: Microsoft Connected Cache is a software-only caching solution that delivers Microsoft content. For more information, see Microsoft Connected Cache overview.
- **MCC Device Count**: The device count is the number of devices that have received bytes from the cache server, for supported content types.
- **Total # of Devices**: The total number of devices with activity in last 28 days.
- **LAN Bytes**: Bytes delivered from LAN peers.
- **Group Bytes**: Bytes from Group peers. If a device is using Group DownloadMode, Delivery Optimization first looks for peers on the LAN and then in the Group. Therefore, if bytes are delivered from LAN peers, they are calculated in 'LAN Bytes'.
- **CDN Bytes**: Bytes delivered from Content Delivery Network (CDN).
- **City**: City is determined based on the location of the device where the maximum amount of data is downloaded.
- **Country**: Country is determined based on the location of the device where the maximum amount of data is downloaded.
- **ISP**: ISP is determined based on the ISP delivering the maximum bytes to the device.

Calculations for Delivery Optimization

The calculated values used in the Delivery Optimization report are listed below.

Efficiency (%) Calculations:

- Bandwidth Savings (BW SAV%) = 100 * (BytesFromPeers + BytesFromGroupPeers + BytesFromCache) / (BytesFromPeers + BytesFromGroupPeers+BytesFromCDN + BytesFromCache)
  - UCDOAggregatedStatus table
- % P2P Efficiency = 100 * (BytesFromPeers + BytesFromGroupPeers) / (BytesFromPeers + BytesFromGroupPeers+BytesFromCDN+BytesFromCache)
  - UCDOStatus table

- % Connected Cache Efficiency = 100 * BytesFromCache / (BytesFromPeers + BytesFromGroupPeers+BytesFromCDN+BytesFromCache)
  - UCDOStatus table

Bytes Calculations:

- TotalBytes = BytesFromCDN + BytesFromEnterpriseCache + BytesFromPeers + BytesFromGroupPeers
  - UCDOAggregatedStatus table
- BytesFromCDN = BytesFromCDN
  - UCDOAggregatedStatus table
- BytesFromPeers = BytesFromLAN
  - UCDOAggregatedStatus table
- BytesFromGroupPeers = BytesFromGroupPeers
  - UCDOAggregatedStatus table
- BytesFromCache = BytesFromCache
  - UCDOAggregatedStatus table

Volume Calculations:

- Volume by P2P = BytesFromPeers + BytesFromGroupPeers
  - UCDOStatus table
- Volume by Connected Cache = BytesFromCache
  - UCDOStatus table
- Volume by CDN = BytesFrom CDN
  - UCDOStatus table

Mapping GroupID

In the **Efficiency By Group** subsection, the **GroupID** is displayed as an encoded SHA256 hash and is case sensitive. You can create a mapping of original to encoded GroupIDs using the following PowerShell example:

PowerShell

```
$text = "<myOriginalGroupID>`0" ; # The `0 null terminator is required

$hashObj = [System.Security.Cryptography.HashAlgorithm]::Create('sha256') ; $dig = $hashObj.ComputeHash([System.Text.Encoding]::Unicode.GetBytes($text)) ; $digB64 = [System.Convert]::ToBase64String($dig) ; Write-Host "$text ==> $digB64"
```

In addition, you can see both the encoded and decoded GroupIDs in the Delivery Optimization logs.

PowerShell

```
Get-DeliveryOptimizationLog -Flush | Set-Content C:\dosvc.log
```

The below two lines are together in verbose logs:

text

```
2023-02-15T12:33:11.3811337Z 1514  1F4          {CGlobalConfigManager::GetGroupId}
Using groupID = **<myOriginalGroupId>**
2023-02-15T12:33:11.3811432Z 1514  1F4          {CGlobalConfigManager::GetGroupId}
Hashed groupID = **<myEncodedGroupId>**
```

Sample queries

You can use the data in UCDOAggregatedStatus and UCDOStatus to create your own queries. Create your custom queries using Kusto Query Language (KQL), but note that Windows Update for Business reports uses Azure Monitor, so some operators aren't supported. The KQL documentation specifies which operators aren't supported by Azure Monitor or if they have different functionality. For more information about KQL in Azure Monitor, see Log queries in Azure Monitor. The following queries are examples of how you can use the data:

Example UCDOAggregatedStatus table query

The following query is used to display the total bandwidth savings % value:

Kusto

```
UCDOAggregatedStatus| where TimeGenerated == _SnapshotTime
| extend LocalSourceBytes = BytesFromCache + BytesFromGroupPeers + BytesFromPeers
| summarize LocalSources_BWSAV = round((sum(0.0 + LocalSourceBytes)/
sum(LocalSourceBytes+BytesFromCDN)) * 100.0 ,2)
| extend Title = "BW SAV%" , SubTitle = "Local Sources"
```

Example UCDOStatus table query

The following query is used to display the Top 10 GroupIDs:

Kusto

```
UCDOStatus  | where TimeGenerated == _SnapshotTime
| summarize sum(BytesFromCDN) ,  sum(BytesFromGroupPeers) , sum(BytesFromPeers) ,
sum(BytesFromCache) ,
DeviceCount = count_distinct(GlobalDeviceId) by GroupID | top 10 by DeviceCount desc
| extend TotalBytes = (sum_BytesFromPeers +
```

```
sum_BytesFromGroupPeers+sum_BytesFromCDN+sum_BytesFromCache)
| extend P2PPercentage = ((0.0 + sum_BytesFromPeers +
sum_BytesFromGroupPeers)/TotalBytes ) * 100.0
| extend  MCCPercentage = ((0.0 + sum_BytesFromCache)/ TotalBytes) * 100.0  ,
 VolumeBytesFromPeers = sum_BytesFromPeers + sum_BytesFromGroupPeers
| extend VolumeBytesFromMCC = sum_BytesFromCache , VolumeByCDN = sum_BytesFromCDN
| project  GroupID , P2PPercentage , MCCPercentage ,  VolumeBytesFromPeers ,
VolumeBytesFromMCC ,VolumeByCDN , DeviceCount
```

Delivery Optimization Supported Content Types

There are many Microsoft content types that are supported by Delivery Optimization. All of these content types show up in the 'Content Distribution' section in the Delivery Optimization report. See the complete table for P2P/Connected Cache support types.

> **Note**
>
> As of July 22, 2025 Software Updates which are non-Windows and non-Store updates will move from the 'Other' category to 'Feature Updates'. Examples include: Visual Studio, Exchange, SQL Server.

Expand table

Content Category	Content Types Included
Apps	Windows 10 Store apps, Windows 11 UWP Store apps
Driver Updates	Windows Update Driver updates
Feature Updates	Windows Update Feature updates
Office	Microsoft 365 Apps and updates
Other	Windows Language Packs, Windows Defender definition updates, Intune Win32 apps, Edge Browser updates, Configuration Manager Express updates, Dynamic updates, MDM Agent, Xbox Game Pass (PC), Windows Package Manager, MSIX Installer (includes Windows 11 Store Win32 apps, Windows 11 Teams updates)
Quality Updates	Windows Updates Quality updates
Teams Updates (P2P support starting with version 25122.1415.3698.6812)	Teams updates

Frequency Asked Questions

- **What time period does the Delivery Optimization data include?** Data is generated/aggregated for the last 28 days for active devices. For Delivery Optimization data to register in the report, the device must have performed some Delivery Optimization action in the 28-day rolling window. This includes device configuration information.

- **Data is showing as 'Unknown', what does that mean?** You may see data in the report listed as 'Unknown'. This status indicates that the Delivery Optimization DownloadMode setting is either invalid or empty.

- **How are the 'Top 10' groups identified?** The top groups are represented by the number of devices in a particular group, for any of the four group types (GroupID, City, Country, and ISP).

- **The GroupIDs don't look familiar, why are they different?** The GroupID values are encoded for data protection telemetry requirements. You can find more information in the 'Mapping GroupIDs' section above.

- **How can I see data for device in the office vs. out of the office?** Today, we don't have a distinction for data that was downloaded by location.

- **What does the data in UCDOStatus table represent?** A row in UCDOStatus represents data downloaded by a combination of a single device ID (AzureADDeviceId) by content type (ContentType).

- **What does the data in UCDOAggregatedStatus table represent?** A row in UCDOAggregatedStatus represents data summarized at the tenant level (AzureADTenantID) for each content type (ContentType).

- **How are BytesFromCache calculated when there's a Connected Cache server used by my ISP?** If there's a Connected Cache server at the ISP level, BytesFromCache filters out any bytes coming the ISP's Connected Cache.

- **How do the results from the Delivery Optimization PowerShell cmdlets compare to the results in the report?** Delivery Optimization PowerShell cmdlets can be a powerful tool used to monitor Delivery Optimization data on the device. These cmdlets use the cache on the device. The data calculated in the report is taken from the Delivery Optimization telemetry events.

- **The report represents the last 28 days of data, why do some queries include >= seven days?** The data in the report does represent the last 28 days of data. The query for last seven days is just to get the data for the latest snapshot from past seven days. It's possible that data is delayed for sometime and not available for current day, so we look for past 7 day snapshot in log analytics and show the latest snapshot.

Last updated on 12/03/2025

Microsoft 365 admin center software updates page

Article • 05/08/2024 •

Applies to: ✅ Windows 11, ✅ Windows 10, ✅ Windows Update for Business reports, ✅ Microsoft 365 admin center

The **Software updates** page in the Microsoft 365 admin center displays a high-level overview of the installation status for Microsoft 365 Apps and Windows updates in your environment. Quality updates that contain security fixes are typically released on the second Tuesday of each month. Ensuring these updates are installed is important because they help protect you from known vulnerabilities. The **Software updates** page allows you to easily determine the overall update compliance for your devices.

The **Software updates** page has following tabs to assist you in monitoring update status for your devices:

- **Microsoft 365 Apps**: Displays update status for Microsoft 365 Apps.
  - For more information about the **Microsoft 365 Apps** tab, see Microsoft 365 Apps updates in the admin center.
- **Windows**: Displays compliance charts for cumulative updates and feature updates for Windows clients. This article contains information about the **Windows** tab.

Limitations

Windows Update for Business reports is a Windows service hosted in Azure that uses Windows diagnostic data. Windows Update for Business reports is available in the Azure Commercial cloud, but not available for GCC High or United States Department of

Defense customers since it doesn't meet US Government community compliance (GCC) requirements. For a list of GCC offerings for Microsoft products and services, see the Microsoft Trust Center.

Get started

After verifying that you've met the prerequisites and permissions for Windows Update for Business reports, enroll using the instructions below if needed:

1. Go to the Microsoft 365 admin center and sign in.
2. Expand **Health**, then select **Software Updates**. You may need to use the **Show all** option to display **Health** in the navigation menu.
   - If you don't see an entry for **Software updates** in the menu, try going to this URL: https://admin.microsoft.com/Adminportal/Home#/softwareupdates .
3. In the **Software Updates** page, select the **Windows** tab.
4. When you select the **Windows** tab for the first time, you'll be asked to **Configure Settings**. This tab is populated by data from Windows Update for Business reports. Verify or supply the following information about the settings for Windows Update for Business reports:
   - The Azure subscription
   - The Log Analytics workspace
5. The initial setup can take up to 24 hours. During this time, the **Windows** tab will display that it's **Waiting for Windows Update for Business reports data**.
   - Enrolling into Windows Update for Business reports doesn't influence the rate that required data is uploaded from devices. Device connectivity to the internet and how active the device is influences how long it will take before the device appears in reporting. Devices that are active and connected to the internet daily can expect to be fully uploaded within one week (usually less than 72 hours). Devices that are less active can take up to two weeks before data is fully available.
6. After the initial setup is complete, the **Windows** tab will display your Windows Update for Business reports data in the charts.

> **Note**

> The device counts in the **Windows** tab may vary from the **Microsoft 365 Apps** tab since their requirements are different.

The Windows tab

The **Windows** tab in the **Software updates** page in the Microsoft admin center is populated by data from Windows Update for Business reports. The tab contains a high-level overview of update compliance for Windows clients in your environment. The tab displays two charts **Windows update status** and **End of service**. The Windows Update for Business reports data that populates these charts refreshes every 24 hours. For more information, see Windows Update for Business reports data latency.

Windows update status chart

The **Windows update status** chart gives you a visual representation of how many devices are in the following states for the monthly cumulative updates:

- Up to date
- Missing security updates
- Unsupported operating system

A device is considered **Up to date** in this chart if it has installed security updates released within the past two months. Devices that are more two months behind on installation are in the **Missing security updates** classification. An **Unsupported operating system** is no longer supported by the Microsoft Product Lifecycle.

End of service chart

The **End of service** chart list the number of devices running an operating system version that's near or past the Microsoft Product Lifecycle. The **End of service** chart lists all operating system versions that aren't the latest version and counts the number of devices for each version. This chart can help you determine how many of your devices need to install the latest operating system feature update. If you're currently deploying feature updates to these devices, the chart can also give you insight into how the deployment is progressing.

End of service

Some devices are reaching end of servicing

Microsoft recommends that you update your devices to the latest version of Windows 10 or upgrade eligible devices to Windows 11 to take advantage of new features and advanced protections from the latest security threats.

Version	Status ↑	Number of devices
Windows 10-20H2	Support ending on 5/9/2023	50
Windows 10-1909	Support ending on 1/10/23	20
Windows 10-1903	Out of support since 12/8/2020	180

Next steps

Use Windows Update for Business reports to display additional data about the status of Windows updates.

Feedback

Was this page helpful? Yes No

Provide product feedback

Use Windows Update for Business reports

Article • 01/19/2023 • Applies to: ✅ Windows 11, ✅ Windows 10

In this article, you'll learn how to use Windows Update for Business reports to monitor Windows updates for your devices. To configure your environment for use with Windows Update for Business reports, see Enable Windows Update for Business reports.

Display Windows Update for Business reports data

1. Sign into the Azure portal.
2. In the Azure portal, type **Log Analytics** in the search bar. As you begin typing, the list filters based on your input.
3. Select **Log Analytics workspaces**.
4. Select the workspace that you use for Windows Update for Business reports.
5. Select **Logs** under the **General** group in your workspace.
6. If the **Always show Queries** option is enabled in Log Analytics, close the query window to access the schema.
7. Under **Schemas and filter**, select **Group by: Solution** and then expand the **Log Management** schema. If the **Group by: Category** is selected, the schema is listed under the **Other** category.
8. Use the Windows Update for Business reports schema for custom Kusto (KQL) queries, to build custom workbooks, or to build your own solution to display the Windows Update for Business reports data. For example, you might query the data to review information for different types of alerts in the past 7 days and how many times each alert occurred.

Kusto

```
UCUpdateAlert
| summarize count=count() by AlertClassification, AlertSubtype, ErrorCode,
Description
```

Data latency

Windows Update for Business reports uses Windows client diagnostic data as its data source. After you add Windows Update for Business reports and appropriately configure your devices, it could take 48-72 hours before they first appear.

The data powering Windows Update for Business reports is refreshed every 24 hours, and refreshes with the latest data from all of your organization's devices that have been seen in the past 28 days. The entire set of data is refreshed in each daily snapshot, which means that the same data can be ingested again even if no new data arrived from the device since the last snapshot. Snapshot time can be determined by the TimeGenerated field for each record, while LastScan can be used to roughly determine the freshness of each record's data. Device connectivity to the internet and generally how active the device is influences how long it will take before it appears in Windows Update for Business reports.

Expand table

Data Type	Data upload rate from device	Data Latency
UCClient	Once per day	4 hours
UCClientUpdateStatus	Every update event (Download, install, etc.)	24-36 hours
UCServiceUpdateStatus	Every update event (Download, install, etc.)	24-36 hours
UCUpdateAlert	Every event	24-36 hours
UCDeviceAlert	Every event	24-36 hours

Data Type	Data upload rate from device	Data Latency
UCClientReadinessStatus	After Windows 11 readiness assessment	24-36 hours
UCDOStatus	Download Events	24-36 hours
UCDOAggregatedStatus	Download Events	24-36 hours

Working with Azure Monitor Logs

Windows Update for Business reports is built on the Azure Monitor Logs platform. All Windows Update for Business reports-related data is collected in a Log Analytics workspace, where the data is available for querying. Understanding the Azure Monitor Logs tools and features at your disposal, all integrated within Azure portal, can deeply enhance your experience and complement Windows Update for Business reports.

See the following Azure Monitor Logs articles to learn how to:

- Query log data effectively in Azure Monitor Logs.
- Create and share dashboards of data in a Log Analytics workspace.
- Set up alerts in Azure Monitor to always stay informed about the critical issues you care about most.

Feedback

Was this page helpful? Yes No

Provide product feedback

Frequently Asked Questions about Windows Update for Business reports

This article answers frequently asked questions about Windows Update for Business reports.

General questions:

- What are Windows Update for Business reports?
- Are Windows Update for Business reports free?
- What Windows versions are supported?

Setup questions:

- How do you set up Windows Update for Business reports?
- Why is "Waiting for Windows Update for Business reports data" displayed on the page
- Why am I getting the error "400 Bad Request: The specified resource already exists"?

Questions about using Windows Update for Business reports:

- Why is the device name null(#)?
- Why am I missing devices in reports?
- What is the difference between OS version and target version?
- Why are there multiple records for the same device?
- Why are devices showing an unknown state?
- When should I use the UCClient, UCClientUpdateStatus, or UCUpdateAlert tables?
- What is the difference between quality and security updates?
- How do I confirm that devices are sending data?
- Why isn't the workbook displaying data even though my UCClient table has data?

Delivery Optimization data:

- What time period does the Delivery Optimization data include?
- Data is showing as "Unknown", what does that mean?
- How are the 'Top 10' groups identified?
- The GroupIDs don't look familiar, why are they different?
- How can I see data for device in the office vs. out of the office?
- What does the data in UCDOStatus table represent?
- What does the data in UCDOAggregatedStatus table represent?
- How are BytesFromCache calculated when there's a Connected Cache server used by my ISP?
- How do the results from the Delivery Optimization PowerShell cmdlets compare to the results in the report?

- The report represents the last 28 days of data, why do some queries include >= seven days?

General

What are Windows Update for Business reports?

Windows Update for Business reports is a cloud-based solution that provides information about your Microsoft Entra joined devices' compliance with Windows updates. Windows Update for Business reports is offered through the Azure portal, and it's included as part of the Windows 10 or Windows 11 prerequisite licenses.

Are Windows Update for Business reports free?

Data ingested into your Log Analytics workspace can be retained at no charge for up to first 31 days (or 90 days if Microsoft Sentinel is enabled on the workspace). Data ingested into Application Insights, either classic or workspace-based, is retained for 90 days without any charge. Data retained beyond these no-charge periods are charged for each GB of data retained for a month, pro-rated daily. For more information, see **Log Data Retention** in Azure Monitor pricing.

What Windows versions are supported?

Windows Update for Business reports supports clients running a supported version of Windows 10 or Windows 11 Professional, Education, Enterprise, and Enterprise multi-session editions. Windows Update for Business reports only provides data for the standard Desktop Windows client version and isn't currently compatible with Windows Server, Surface Hub, IoT, or other versions.

> **Important**
>
> Currently there is a known issue where Windows Update for Business reports doesn't display data for Enterprise multi-session edition devices.

Setup questions

How do you set up Windows Update for Business reports?

After verifying the prerequisites are met, you can start to set up Windows Update for Business reports. The two main steps for setting up Windows Update for Business reports are:

1. Add Windows Update for Business reports to your Azure subscription. This step has the following phases:
2. Select or create a new Log Analytics workspace for use with Windows Update for Business reports.
3. Enroll into Windows Update for Business reports using one of the following methods:

- Enroll through the Azure Workbook (preferred method) - Enroll from the Microsoft 365 admin center.

1. Configure the clients to send data to Windows Update for Business reports. You can configure clients in the following three ways:

    - Use a script
    - Use Microsoft Intune
    - Configure manually

Why is `Waiting for Windows Update for Business reports data` displayed on the page?

Typically, the **Waiting for Windows Update for Business reports data** message is displayed because:

- You may not have the correct permissions to display the data.
- The initial enrollment may not be complete yet.
- It's possible that devices aren't sharing data. If you received a successful save message during enrollment but still haven't seen any data after 48 hours, try using the configuration script on devices to ensure they're configured properly. If you've verified the above items, but still aren't seeing data, you can unenroll then re-enroll. However, it takes another 24-48 hours for the enrollment to complete. If the issue persists, contact support.

Why am I getting the error `400 Bad Request: The specified resource already exists`?

A `400 Bad Request: The specified resource already exists` error message indicates that the service already has a subscription and workspace mapping saved. If you're trying to re-enroll with the same configuration settings, wait a few minutes, then refresh the page before saving your subscription and workspace again. Sometimes it can take time to register the save, so it's important to not re-enroll too quickly.

Using Windows Update for Business reports

Why is the device name null(#)?

If you're seeing the device ID but not the device name, it's possible that the required policy for displaying the device name isn't set on the client. Ensure clients have the policy configured.

- CSP: System/AllowDeviceNameInDiagnosticData
- Group Policy: Allow device name to be sent in Windows diagnostic data
  - Located in **Computer Configuration** > **Administrative Templates** > **Windows Components** >**Data Collection and Preview Builds**. It can take up to 21 days for all device names to show in up in reports assuming they're powered on and active.

Why am I missing devices in reports?

Here are some reasons why you may not be seeing devices in reports:

- **The device isn't enrolled with Microsoft Entra**: A prerequisite for devices is that they're either Microsoft Entra joined or hybrid Microsoft Entra joined.
- **The device isn't sending data**: It's possible devices aren't sharing data due to a policy being incorrectly configured or a proxy or firewall configuration. Try using the configuration script on devices to ensure they're configured properly.
- **The device isn't active enough**: Clients must be active and connected to the internet to scan against Microsoft Update. Ensure devices are powered on and have been active at least once in the past 28 days.
- **The workbook has limited the results**: The default limit for rows in Azure workbooks is set to 1000. This limit is to avoid any delay in the load time for the interface. If you noticed that you can't find a specific device, you can export the output in Excel, or open the results in the logs view for the full result by selecting the three dots beside each component.

Why are there multiple records for the same device?

Devices have multiple records when the `UCClientUpdateStatus` or `UCClientServiceStatus` tables are queried. These tables contain multiple records because they have the history for all devices that have discovered applicable updates within the past 28 days. For example, it's possible that a device has discovered multiple security updates, each with different update states, at various times over the past 28 days. It's also possible that a device can be in multiple deployments, so multiple records are displayed.

Why are devices showing an unknown state?

An unknown client state is displayed if there isn't an update record for the device. This state can happen for many reasons, like the device not being active, not being able to scan Windows Update, or it doesn't currently have any update related activity occurring.

What is the difference between OS version and target version?

The word *target* in data labels refers to the update version, build, or KB the client intends to update to. Typically, the fields starting with *OS*, such as OSbuild and OSversion, represents what the device is currently running.

When should I use the UCClient, UCClientUpdateStatus, or UCUpdateAlert tables?

These tables can be used for the following information:

- **UCClient**: Represents an individual device's record. It contains data such as the device's name, currently installed build, and the OS Edition. Each device has one record in this table. Use this table to get the overall compliance status of your devices.
  - To display information for a specific device by Microsoft Entra device ID: `UCClient where AzureADDeviceId contains "01234567-89ab-cdef-0123-456789abcdef"`
  - To display all device records for devices running any Windows 11 OS version: `UCClient | where OSVersion contains "Windows 11"`

- **UCClientUpdateStatus**: Contains records for every update the device determined was applicable. There can be multiple records for a device if it's discovered multiple applicable updates in the past 60 days. Use this table if you want to get detailed update status for your active deployments. There will typically be 3 update status records per device for the latest 3 security updates.
  - To find device records for devices that determined the March 14, 2023 update was applicable: `UCClientUpdateStatus | where UpdateCategory =="WindowsQualityUpdate" and UpdateReleaseTime == "3/14/2023"`
  - To display devices that are in the restart required substate: `UCClientUpdateStatus |where ClientSubstate =="RestartRequired"`

- **UCUpdateAlert**: Use this table to understand update failures and act on devices through alert recommendations. This table contains information that needs attention, relative to

one device, one update, and one deployment (if relevant).

- To display information about an error code: `UCUpdateAlert|where ErrorCode =="0X8024000b"`
- To display a count of devices with active alerts by subtype: `UCUpdateAlert |where AlertStatus =="Active"|summarize Devices=count() by AlertSubtype`

What is the difference between quality and security updates?

Windows quality updates are monthly updates that are released on the second or fourth Tuesday of the month. The cumulative updates released on the second Tuesday of the month can contain both security updates and nonsecurity updates. Cumulative updates released on the fourth Tuesday of the month are optional nonsecurity preview releases. Use the fields within the UCClient table for additional information, such as:

- **OSSecurityUpdateStatus**: Indicates the status of the monthly update that's released on the second Tuesday
- **OSQualityUpdateStatus**: Indicates the status of the monthly update that's released on the fourth Tuesday

How do I confirm that devices are sending data?

Once enrollment is done and devices are properly configured to share data, wait for 48 hours for data to start showing up in reports. It can take up to 14 days for all of your devices to show up in reports in some cases where devices aren't active much. You can check to see if the Log Analytics tables are being populated in your workspace. The data is ingested by the service daily to generate reports. If you notice a day is missing, it's possible that the reports service missed an ingestion. To confirm devices are sending data, query the UCClient table. The following query shows total enrolled device count per time-generated:

```
UCClient | summarize count() by TimeGenerated
```

Why isn't the workbook displaying data even though my UCClient table has data?

If the UCClient table has data, but the workbook isn't displaying data, ensure that the user has correct permissions to read the data. The Log Analytics Reader role is needed to view the data in the workbooks. The Log Analytics Contributor role is needed to do any edits to the queries and workbooks.

Delivery Optimization data

What time period does the Delivery Optimization data include?

Data is aggregated for the last 28 days for active devices.

Data is showing as 'Unknown', what does that mean?

You may see data in the report listed as 'Unknown'. This status indicates that the Delivery Optimization DownloadMode setting is either invalid or empty.

How are the 'Top 10' groups identified?

The top groups are represented by the number of devices in a particular group, for any of the four group types (GroupID, City, Country, and ISP).

The GroupIDs don't look familiar, why are they different?

The GroupID values are encoded for data protection requirements. For more information, see Mapping GroupIDs.

How can I see data for device in the office vs. out of the office?

Today, we don't have a distinction for data that was downloaded by location.

What does the data in UCDOStatus table represent?

A row in UCDOStatus represents data downloaded by a combination of a single device ID (AzureADDeviceId) by content type (ContentType).

What does the data in UCDOAggregatedStatus table represent?

A row in UCDOAggregatedStatus represents data summarized at the tenant level (AzureADTenantID) for each content type (ContentType).

How are BytesFromCache calculated when there's a Connected Cache server used by my ISP?

If there's a Connected Cache server at the ISP level, BytesFromCache filters out any bytes coming the ISP's Connected Cache.

How do the results from the Delivery Optimization PowerShell cmdlets compare to the results in the report?

Delivery Optimization PowerShell cmdlets can be a powerful tool used to monitor Delivery Optimization data on the device. These cmdlets use the cache on the device. The data calculated in the report is taken from the Delivery Optimization events.

The report represents the last 28 days of data, why do some queries include >= seven days?

The data in the report does represent the last 28 days of data. The query for last seven days is just to get the data for the latest snapshot from past seven days. It's possible that data is delayed for sometime and not available for current day, so we look for past seven day snapshot in log analytics and show the latest snapshot.

Windows Update for Business reports feedback, support, and troubleshooting

Article • 07/09/2024 • Applies to: ✅ Windows 11, ✅ Windows 10

There are several resources that you can use to find help with Windows Update for Business reports. Whether you're just getting started or an experienced administrator, use the following resources when you need help with Windows Update for Business reports:

- Send product feedback about Windows Update for Business reports
- Open a Microsoft support case
- Documentation feedback
- Follow the Windows IT Pro blog to learn about upcoming changes to Windows Update for Business reports
- Use Microsoft Q&A to ask product questions

Send product feedback

Use the product feedback option to offer suggestions for new features and functionality, or to suggest changes to the current Windows Update for Business reports features. You can share feedback directly to the Windows Update for Business reports product group. To provide product feedback:

1. In the upper right corner of the Azure portal, select the feedback icon.
2. Select either the smile or the frown to rate your satisfaction with your experience.
3. In the text box, describe what you did or didn't like. When providing feedback about a problem, be sure to include enough detail in your description so it can be properly identified by the product group.
4. Choose if you'd like to allow Microsoft to email you about your feedback.

5. Select **Submit feedback** when you've completed the feedback form.

Open a Microsoft support case

You can open support requests directly from the Azure portal. If the **Help + Support** page doesn't display, verify you have access to open support requests. For more information about role-based access controls for support requests, see Create an Azure support request. To create a new support request for Windows Update for Business reports:

1. Open the **Help + Support** page from the following locations:

- In the Send product feedback flyout, select the **contact support** link.
- From the Azure portal, select **New support request** under the **Support + Troubleshooting** heading.

1. Select **Create a support request**, which opens the new support request page.

2. On the **Problem description** tab, provide information about the issue. The following items in ***bold italics*** should be used to help ensure a Windows Update for Business reports engineer receives your support request:

    - **Summary** - Brief description of the issue
    - **Issue type** - ***Technical***
    - **Subscription** - Select the subscription used for Windows Update for Business reports
    - **Service** - ***My services***
    - **Service type** - Select ***Windows Update for Business reports*** under ***Monitoring and Management***

3. Based on the information you provided, you are shown some **Recommended solutions** you can use to try to resolve the problem.

4. Complete the **Additional details** tab and then create the request on the **Review + create** tab.

Documentation feedback

Select the **Feedback** link in the upper right of any article to go to the Feedback section at the bottom. Feedback is integrated with GitHub Issues. For more information about this integration with GitHub Issues, see the docs platform blog post.

To share feedback about the current article, select **This page**. A GitHub account is a prerequisite for providing documentation feedback. Once you sign in, there's a one-time authorization for the MicrosoftDocs organization. It then opens the GitHub new issue form. Add a descriptive title and detailed feedback in the body, but don't modify the document details section. Then select **Submit new issue** to file a new issue for the target article in the Windows-ITPro-docs GitHub repository.

To see whether there's already feedback for this article, select **View all page feedback**. This action opens a GitHub issue query for this article. By default it displays both open and closed issues. Review any existing feedback before you submit a new issue. If you find a related issue, select the face icon to add a reaction, add a comment to the thread, or **Subscribe** to receive notifications.

Use GitHub Issues to submit the following types of feedback:

- Doc bug: The content is out of date, unclear, confusing, or broken.
- Doc enhancement: A suggestion to improve the article.
- Doc question: You need help with finding existing documentation.
- Doc idea: A suggestion for a new article.
- Kudos: Positive feedback about a helpful or informative article.
- Localization: Feedback about content translation.
- Search engine optimization (SEO): Feedback about problems searching for content. Include the search engine, keywords, and target article in the comments.

If you create an issue for something not related to documentation, Microsoft will close the issue and redirect you to a better feedback channel. For example:

- Product feedback for Windows Update for Business reports
- Product questions (using Microsoft Q&A)
- Support requests for Windows Update for Business reports

To share feedback about the Microsoft Learn platform, see Microsoft Learn feedback. The platform includes all of the wrapper components such as the header, table of

contents, and right menu. Also how the articles render in the browser, such as the font, alert boxes, and page anchors.

Feedback

Was this page helpful? Yes No

Provide product feedback

Windows Update for Business reports schema

Article • 12/06/2023 • Applies to: ✅ Windows 11, ✅ Windows 10

When the visualizations provided in the default experience don't fulfill your reporting needs, or if you need to troubleshoot issues with devices, it's valuable to understand the schema for Windows Update for Business reports and have a high-level understanding of the capabilities of Azure Monitor log queries to power additional dashboards, integration with external data analysis tools, automated alerting, and more.

Schemas for Windows Update for Business reports

The following table summarizes the different tables that are part of the Windows Update for Business reports solution. To learn how to navigate Azure Monitor Logs to find this data, see Get started with log queries in Azure Monitor.

> **Note**
>
> Data is collected daily. The TimeGenerated field shows the time data was collected. It's added by Log Analytics when data is collected. Device data from the past 28 days is collected, even if no new data has been generated since the last time. LastScan is a clearer indicator of data freshness (that is, the last time the values were updated), while TimeGenerated indicates the freshness of data within Log Analytics.

Expand table

Table	Category	Description
UCClient	Device record	UCClient acts as an individual device's record. It contains data such as the currently installed build, the device's name, the operating system edition, and active hours (quantitative).
UCClientReadinessStatus	Device record	UCClientReadinessStatus is an individual device's record about its readiness for updating to Windows 11. If the device isn't capable of running Windows 11, the record includes which Windows 11 hardware requirements the device doesn't meet.

Table	Category	Description
UCClientUpdateStatus	Device record	Update Event that combines the latest client-based data with the latest service-based data to create a complete picture for one device (client) and one update.
UCDeviceAlert	Service and device record	These alerts are activated as a result of an issue that is device-specific. It isn't specific to the combination of a specific update and a specific device. Like UpdateAlerts, the AlertType indicates where the Alert comes from such as a ServiceDeviceAlert or ClientDeviceAlert.
UCDOAggregatedStatus	Device record	UCDOAggregatedStatus is an aggregation of all individual UDDOStatus records across the tenant and summarizes bandwidth savings across all devices enrolled using Delivery Optimization and Microsoft Connected Cache.
UCDOStatus	Device record	UCDOStatus provides information, for a single device, on its bandwidth utilization across content types in the event they use Delivery Optimization and Microsoft Connected Cache.
UCServiceUpdateStatus	Service record	Update Event that comes directly from the service-side. The event has only service-side information for one device (client), and one update, in one deployment.
UCUpdateAlert	Service and device records	Alert for both client and service update. Contains information that needs attention, relative to one device (client), one update, and one deployment, if relevant. Certain fields may be blank depending on the UpdateAlert's AlertType field. For example, ServiceUpdateAlert won't necessarily contain client-side statuses and may be blank.

Feedback

Was this page helpful? Yes No

Provide product feedback

UCClient

Article • 03/12/2024 • Applies to: ✅ Windows 11, ✅ Windows 10

UCClient acts as an individual device's record. It contains data such as the currently installed build, the device's name, the OS edition, and active hours (quantitative).

Schema for UCClient

Expand table

Field	Type	Enumerated type	Example	Description
AzureADDeviceID	string	No	`71db1a1a-f1a6-4a25-b88f-79c2f513dae0`	Microsoft Entra Device ID
AzureADTenantId	string	No	`69ca04b0-703d-4b3a-9184-c4e3c15d6f5e`	Microsoft Entra tenant ID
City	string	No		Currently, data isn't gathered to populate this field. Device city, based on IP address.
Country	string	No	`US`	The last-reported location of device (country or region), based on IP address. Shown as country code.
DeviceFamily	string	No	`PC, Phone`	The device family such as PC, Phone.
DeviceFormFactor	string	No	`Notebook, Desktop, Phone.`	Currently, data isn't gathered to populate this field. The device form factor
DeviceManufacturer	string	No	`Hewlett-Packard.`	Currently, data isn't gathered to populate this field. The device OEM manufacturer
DeviceModel	string	No	`The device's OEM model`	Currently, data isn't gathered to populate this field. The device OEM model
DeviceName	string	No	`JohnPC-Contoso`	Client-provided device name
GlobalDeviceId	string	No	`g:9832741921341`	The global device identifier
IsVirtual	bool	No	`Yes, No`	Whether device is a virtual device.
LastCensusScanTime [UTC]	datetime	No	`2020-05-14 09:26:03.478039`	The last time this device performed a successful census scan, if any.
LastWUScanTime [UTC]	datetime	No	`2020-05-14 09:26:03.478039`	The last time this device performed a successful Windows Update scan, if any.
OSArchitecture	string	Yes	`x86`	The architecture of the operating system (not the device) this device is currently on.
OSBuild	string	No	`10.0.22621.1702`	The full operating system build installed on this device, such as Major.Minor.Build.Revision
OSBuildNumber	int	No	`22621`	The major build number, in int format, the device is using.
OSEdition	string	Yes	`Professional`	The Windows edition
OSFeatureUpdateComplianceStatus	string	Yes	`Compliant`	Whether the device is on the latest feature update that's offered from Windows Autopatch, else NotApplicable.

Field	Type	Enumerated type	Example	Description
OSFeatureUpdateEOSTime [UTC]	datetime	No	`2020-05-14 09:26:03.478039`	The end of service date of the feature update currently installed on the device.
OSFeatureUpdateReleaseTime [UTC]	datetime	No	`2020-05-14 09:26:03.478039`	The release date of the feature update currently installed on the device.
OSFeatureUpdateStatus	string	Yes	`InService;EndOfService`	Whether the device is on the latest available feature update, for its feature update.
OSQualityUpdateComplianceStatus	string	Yes	`NotCompliant`	Whether the device is on the latest quality update that's offered from Windows Autopatch, else NotApplicable.
OSQualityUpdateReleaseTime [UTC]	datetime	No	`2020-05-14 09:26:03.478039`	The release date of the quality update currently installed on the device.
OSQualityUpdateStatus	string	Yes	`Latest;NotLatest`	Whether the device is on the latest available quality update, for its feature update.
OSRevisionNumber	int	No	`836`	The revision, in int format, this device is on.
OSSecurityUpdateComplianceStatus	string	Yes	`NotCompliant`	Whether the device is on the latest security update (quality update where the Classification=Security) that's offered from Windows Autopatch, else NotApplicable.
OSSecurityUpdateStatus	string	Yes	`Latest;NotLatest;MultipleSecurityUpdatesMissing`	Whether the device is on the latest available security update, for its feature update.
OSServicingChannel	string	Yes	`SAC`	The elected Windows 10 servicing channel of the device.
OSVersion	string	No	`1909`	The Windows 10 operating system version currently installed on the device, such as 19H2, 20H1, 20H2.
PrimaryDiskFreeCapacityMb		No		Currently, data isn't gathered to populate this field. Free disk capacity of the primary disk in Megabytes.
SCCMClientId	string	No	`5AB72FAC-93AB-4954-9AB0-6557D0EFA245`	Configuration Manager client ID, if available.
SourceSystem	string	No	`Azure`	
TenantId	string	No	`69ca04b0-703d-4b3a-9184-c4e3c15d6f5e`	Microsoft Entra tenant ID of the device.
TimeGenerated [UTC]	datetime	No	`2020-05-14 09:26:03.478039`	The time the snapshot generated this specific record. This field is to determine to which batch snapshot this record belongs.
Type	string	No	`UCClient`	The entity type
UpdateConnectivityLevel		Yes		Currently, data isn't gathered to populate this field. Whether or not this device is maintaining a sufficiently cumulative and continuous connection to Windows Update so the update can progress optimally.

Field	Type	Enumerated type	Example	Description
WUAutomaticUpdates		No		Currently, data isn't gathered to populate this field. Manage automatic update behavior to scan, download, and install updates.
WUDeadlineNoAutoRestart		No		Currently, data isn't gathered to populate this field. Devices won't automatically restart outside of active hours until the deadline is reached - It's 1 by default and indicates enabled, 0 indicates disabled
WUDODownloadMode		No		Currently, data isn't gathered to populate this field. The Windows Update DO DownloadMode configuration.
WUFeatureDeadlineDays	int	No	`0`	CSP: ConfigureDeadlineForFeatureUpdates. The feature update deadline configuration in days. -1 indicates not configured. 0 indicates configured but set to 0. Values > 0 indicate the deadline in days.
WUFeatureDeferralDays	int	No	`0`	CSP: DeferFeatureUpdates. The feature update deferral configuration in days. -1 indicates not configured. 0 indicates configured but set to 0. Values > 0 indicate the policy setting.
WUFeatureGracePeriodDays	int	No	`7`	The Windows Update grace period for feature update in days. -1 indicates not configured. 0 indicates configured and set to 0. Values greater than 0 indicate the grace period in days.
WUFeaturePauseEndTime [UTC]	datetime	No	`2020-05-14 09:26:03.478039`	Currently, data isn't gathered to populate this field. The time Windows Update feature update pause ends, if activated, else null.
WUFeaturePauseStartTime [UTC]	datetime	No	`2020-05-14 09:26:03.478039`	Currently, data isn't gathered to populate this field. The time Windows Update feature update pause was activated, if activated, else null. Feature updates are paused for 35 days from the specified start date.
WUFeaturePauseState	string	No	`NotConfigured`	Indicates pause status of device for feature updates. Possible values are Paused, NotPaused, NotConfigured.
WUNotificationLevel		No		Currently, data isn't gathered to populate this field. This policy allows you to define what Windows Update notifications users see. 0 (default) - Use the default Windows Update notifications. 1 - Turn off all notifications, excluding restart warnings. 2 - Turn off all notifications, including restart warnings
WUPauseUXDisabled		No		Currently, data isn't gathered to populate this field. This policy allows the IT admin to disable the Pause Updates feature. When this policy is enabled, the user can't access the Pause updates' feature. Supported values 0, 1.

Field	Type	Enumerated type	Example	Description
WUQualityDeadlineDays	int	No	`7`	CSP: ConfigureDeadlineForQualityUpdates. The Windows update quality update deadline configuration in days. -1 indicates not configured. 0 indicates configured but set to 0. Values > 0 indicate the deadline in days.
WUQualityDeferralDays	int	No	`-1`	CSP: DeferQualityUpdates. The Windows Update quality update deferral configuration in days. -1 indicates not configured. 0 indicates configured but set to 0. Values greater than 0 indicate the policy setting.
WUQualityGracePeriodDays	int	No	`0`	The Windows Update grace period for quality update in days. -1 indicates not configured. 0 indicates configured and set to 0. Values greater than 0 indicate the grace period in days.
WUQualityPauseEndTime [UTC]	datetime	No	`2020-05-14 09:26:03.478039`	Currently, data isn't gathered to populate this field. The time the quality update pause ends, if activated, else null.
WUQualityPauseStartTime [UTC]	datetime	No	`2020-05-14 09:26:03.478039`	Currently, data isn't gathered to populate this field. The time Windows Update quality update pause- was activated; if activated; else null.
WUQualityPauseState	string	No	`NotConfigured`	Indicates pause status of device for quality updates. Possible values are Paused, NotPaused, NotConfigured.
WURestartNotification		No		Currently, data isn't gathered to populate this field. Allows the IT Admin to specify the method by which the auto restart required notification is dismissed. The following list shows the supported values: 1 (default) = Auto Dismissal. 2 - User Dismissal.
WUServiceURLConfigured		No		Currently, data isn't gathered to populate this field. The device checks for updates from Microsoft Update. Set to a URL, such as http://abcd-srv:8530. The device checks for updates from the WSUS server at the specified URL. Not configured. The device checks for updates from Microsoft Update. Set to a URL, such as http://abcd-srv:8530. The device checks for updates from the WSUS server at the specified URL.
WUUXDisabled		No		Currently, data isn't gathered to populate this field. This policy allows the IT admin to remove access to scan Windows Update. When this policy is enabled, the user can't access the Windows Update scan, download, and install features. Default is 0. Supported values 0, 1.

Feedback

Was this page helpful? Yes No

Provide product feedback

UCClientReadinessStatus

Article • 12/06/2023 • Applies to: ✅ Windows 11, ✅ Windows 10

UCClientReadinessStatus is an individual device's record about its readiness for updating to Windows 11. If the device isn't capable of running Windows 11, the record includes which Windows 11 hardware requirements the device doesn't meet.

Schema for UCClientReadinessStatus

Expand table

Field	Type	Enumerated type	Example	Description
AzureADDeviceId	string	No	`71db1a1a-f1a6-4a25-b88f-79c2f513dae0`	Microsoft Entra Device ID
AzureADTenantId	string	No	`69ca04b0-703d-4b3a-9184-c4e3c15d6f5e`	Microsoft Entra tenant ID
DeviceName	string	No	`JohnPC-Contoso`	Client-provided device name
GlobalDeviceId	string	No	`g:9832741921341`	The global device identifier.
OSBuild	string	No	`10.0.18363.836`	The full OS build installed on this device, such as Major.Minor.Build.Revision
OSName	string	No	`Windows 10`	The operating system name.
OSVersion	string	No	`1909`	The Win10 OS version (such as 19H2, 20H1, 20H2) currently installed on the device.
ReadinessExpiryTime [UTC]	datetime	No	`2020/05/14 09:26:03.478 AM`	The date and time when the readiness assessment will expire.
ReadinessReason	string	No	`CPU;TPM`	Lists which hardware requirements are blocking the device from being capable of installing Windows 11. Field is null if the device is capable. This status is determined by

Field	Type	Enumerated type	Example	Description
				the Windows Update applicability.
ReadinessScanTime [UTC]	datetime	No	`2020/05/14 09:26:03.478 AM`	The date and time when readiness was assessed and the assessment was sent.
ReadinessStatus	string	No	`Not capable`	The readiness status of the device is either capable, not capable, or unknown. This status is determined by Windows Update.
SCCMClientId	string	No	`5AB72FAC-93AB-4954-9AB0-6557D0EFA245`	Configuration Manager Client ID, if available.
SetupReadinessExpiryTime [UTC]	datetime	No	`2020/05/14 09:26:03.478 AM`	The date and time when the setup readiness assessment will expire.
SetupReadinessReason	string	No	`CPU;TPM`	Lists which hardware requirements are blocking the device from being capable of installing Windows 11. Field is null if the device is capable. This status is determined by Windows setup.
SetupReadinessStatus	string	No	`Not capable`	The readiness status of the device is either capable, not capable, or unknown. This status is determined by Windows setup.
SetupReadinessTime [UTC]	datetime	No	`2020/05/14 09:26:03.478 AM`	The date and time when readiness was assessed by setup and the assessment was sent.
SourceSystem	string	No	`Azure`	
TargetOSBuild	string	No	`10.0.22000.1`	The full operating system build number that's being targeted to the device for this readiness record.

Field	Type	Enumerated type	Example	Description
TargetOSName	string	No	`Windows 11`	The name of the operating system being targeted to the device for this readiness record.
TargetOSVersion	string	No	`21H2`	The operating system version being targeted to the device for this readiness record.
TenantId	string	No	`69ca04b0-703d-4b3a-9184-c4e3c15d6f5e`	Microsoft Entra tenant ID of the device.
TimeGenerated [UTC]	datetime	No	`2020/05/14 09:26:03.478 AM`	The date and time when Azure Monitor Logs ingested this record for your Log Analytics workspace.
Type	string	No	`UCClientReadinessStatus`	The entity type

Feedback

Was this page helpful? Yes No

Provide product feedback

UCClientUpdateStatus

Article • 12/06/2023 • Applies to: ✅ Windows 11, ✅ Windows 10

Update Event that combines the latest client-based data with the latest service-based data to create a complete picture for one device (client) and one update.

Schema for UCClientUpdateStatus

Expand table

Field	Type	Enumerated type	Example	Description
AzureADDeviceId	string	No	`71db1a1a-f1a6-4a25-b88f-79c2f513dae0`	Microsoft Entra Device ID
AzureADTenantId	string	No	`69ca04b0-703d-4b3a-9184-c4e3c15d6f5e`	Microsoft Entra tenant ID
CatalogId	string	No	`b0f410599615e2ce15e6614ac3fc4ec62d80324020351e172edef89091a64f2f`	This field applies to drivers only. The Catalog ID of the update from Windows Autopatch.
ClientState	string	Yes	`Installing`	This field applies to drivers only. Higher-level bucket of ClientSubstate.
ClientSubstate	string	Yes	`DownloadStart`	Last-known state of this update relative to the device, from the client.
ClientSubstateRank	int	No	`2300`	Ranking of client substates for sequential ordering in funnel-type views. The rankings between ServiceSubstate and ClientSubstate can be used together.
ClientSubstateTime [UTC]	datetime	No	`2020/05/14 09:26:03.478 AM`	Date and time of last client substate transition
DeploymentId	string	No	`cf1b12a3-3d84-4ce3-bc8e-de48459e252d`	The identifier of the deployment that is targeting this update to this device, else empty.
DeviceName	string	No	`JohnPC-Contoso`	Device's given name
EventData	string	No		Currently, data isn't gathered to populate this field. Json to fill with arbitrary K/V pairs. Used to populate contextual data that would otherwise be

Field	Type	Enumerated type	Example	Description
				sparsely populated if elevated to a field always present in the schema.
FurthestClientSubstate	string	No	`DownloadComplete`	Furthest clientSubstate
FurthestClientSubstateRank	int	No	`2400`	Ranking of furthest clientSubstate
GlobalDeviceId	string	No	`g:9832741921341`	Microsoft internal global device identifier
IsUpdateHealthy	string	No	`1`	Currently, data isn't gathered to populate this field. True: No issues preventing this device from updating to this update have been found. False: There's something that may prevent this device from updating.
OfferReceivedTime [UTC]	datetime	No	`2020/05/14 09:26:03.478 AM`	Date and time when device last reported entering OfferReceived, else empty.
RestartRequiredTime [UTC]	datetime	No	`2020/05/14 09:26:03.478 AM`	Date and time when device first reported entering RebootRequired (or RebootPending), else empty.
SCCMClientId	string	No	`5AB72FAC-93AB-4954-9AB0-6557D0EFA245`	A string corresponding to the Configuration Manager Client ID on the device.
SourceSystem	string	No	`Azure`	
TargetBuild	string	No	`10.0.18363.836`	The full build of the content this DeviceUpdateEvent is tracking. For Windows 10 updates, this value would correspond to the full build (10.0.14393.385).
TargetBuildNumber	string	No	`18363`	Integer of the Major portion of Build.
TargetKBNumber	string	No	`KB4524570`	KB Article.
TargetRevisionNumber	string	No	`836`	Integer or the minor (or revision) portion of the build.

Field	Type	Enumerated type	Example	Description
TargetVersion	string	No	`1909`	The target operating system version, such as 1909.
TenantId	string	No	`69ca04b0-703d-4b3a-9184-c4e3c15d6f5e`	Microsoft Entra tenant ID of the device.
TimeGenerated [UTC]	datetime	No	`2020/05/14 09:26:03.478 AM`	The time the snapshot generated this specific record. This is to determine to which batch snapshot this record belongs.
Type	string	No	`UCClientUpdateStatus`	The entity type
UpdateCategory	string	Yes	`WindowsFeatureUpdate`	The type of content this DeviceUpdateEvent is tracking.
UpdateClassification	string	Yes	`Upgrade`	Whether the update classification is an upgrade (feature update), security (quality update), nonsecurity (quality update), or driver
UpdateConnectivityLevel		Yes		Currently, data isn't gathered to populate this field. Whether or not this device is maintaining a sufficiently cumulative and continuous connection to Windows Update so the update can progress optimally.
UpdateDisplayName	string	No	`Windows 10 1909`	The long-form display name for the given update. Varies on content type (feature update. quality update)
UpdateHealthGroupL1	string	No		Currently, data isn't gathered to populate this field. Grouping design to describe the current update installation's "health", L1 (highest-level).
UpdateHealthGroupL2	string	No		Currently, data isn't gathered to populate this field. Integer for ranking

Field	Type	Enumerated type	Example	Description
				the L1 UpdateHealthGroup.
UpdateHealthGroupL3	string	No		Currently, data isn't gathered to populate this field. Second grouping, subset of L1, more detailed.
UpdateHealthGroupRankL1	int	No		Currently, data isn't gathered to populate this field. Integer for ranking the L2 UpdateHealthGroup.
UpdateHealthGroupRankL2	int	No		Currently, data isn't gathered to populate this field. Third grouping, subset of L3, more detailed.
UpdateHealthGroupRankL3	int	No		Currently, data isn't gathered to populate this field. Integer for ranking the L3 UpdateHealthGroup.
UpdateId	string	No	`10e519f0-06ae-4141-8f53-afee63e995f0`	This field applies to drivers only. Update ID of the targeted update
UpdateInstalledTime [UTC]	datetime	No	`2020/05/14 09:26:03.478 AM`	DateTime when event transitioned to UpdateInstalled, else empty.
UpdateManufacturer	string	No	`Microsoft`	This field applies to drivers only. Manufacturer of update. Microsoft for feature or quality updates, for drivers the name of driver manufacturer.
UpdateReleaseTime [UTC]	datetime	No	`2020/05/14 09:26:03.478 AM`	The release date of the update
UpdateSource	string	Yes	`UUP`	The source of the update such as UUP, MUv6, Media

Feedback

Was this page helpful? Yes No

Provide product feedback

UCDeviceAlert

Article • 12/06/2023 • Applies to: ✅ Windows 11, ✅ Windows 10

These alerts are activated as a result of an issue that is device-specific. It isn't specific to the combination of a specific update and a specific device. Like UpdateAlerts, the AlertType indicates where the Alert comes from (ServiceDeviceAlert, ClientDeviceAlert). For example, an EndOfService alert is a ClientDeviceAlert, as a build no longer being serviced (EOS) is a client-wide state. Meanwhile, DeviceRegistrationIssues in Windows Autopatch will be a ServiceDeviceAlert, as it's a device-wide state in the service to not be correctly registered.

Schema for UCDeviceAlert

Expand table

Field	Type	Enumerated type	Example	Description
AlertClassification	string	Yes	`Error`	Whether this alert is an Error, a Warning, or Informational
AlertData	string	No		Currently, data isn't gathered to populate this field. An optional string formatted as a json payload containing metadata for the alert.
AlertId	string	No	`9e107d9d372bb6826bd81d3542a419d6`	The unique identifier of this alert
AlertRank	int	No	`1000`	Integer ranking of alert for prioritization during troubleshooting
AlertStatus	string	Yes	`Active`	Whether this alert is Active, Resolved, or Deleted
AlertSubtype	string	Yes	`DiskFull`	The subtype of alert.
AlertType	string	Yes	`ClientUpdateAlert`	The type of alert such as ClientUpdateAlert

Field	Type	Enumerated type	Example	Description
				or ServiceUpdateAlert. Indicates which fields are present.
AzureADDeviceId	string	No	`71db1a1a-f1a6-4a25-b88f-79c2f513dae0`	Microsoft Entra Device ID
AzureADTenantId	string	No	`69ca04b0-703d-4b3a-9184-c4e3c15d6f5e`	Microsoft Entra tenant ID
Description	string	No	`Disk full`	A localized string translated from a combination of other alert fields + language preference that describes the issue in detail.
DeviceName	string	No	`JohnPC-Contoso`	The given device's name
ErrorCode	string	No		Currently, data isn't gathered to populate this field. The Error Code, if any, that triggered this Alert. In the case of Client-based explicit alerts, error codes can have extended error codes, which are appended to the error code with an underscore separator.
ErrorSymName	string	No		Currently, data isn't gathered to populate this field. The symbolic name that maps to the Error Code, if any. Otherwise empty.
GlobalDeviceId	string	No	`g:1298371934870`	Internal Microsoft global identifier, if available.

Field	Type	Enumerated type	Example	Description
Recommendation	string	No	`Free up disk space.`	A localized string translated from RecommendedAction, Message, and other fields (depending on source of alert) that provides a recommended action.
ResolvedTime [UTC]	datetime	No	`2020-05-14 09:26:03.478039`	The time this alert was resolved, else empty.
SCCMClientId	string	No	`5AB72FAC-93AB-4954-9AB0-6557D0EFA245`	Configuration Manager client ID of the device, if available.
SourceSystem	string	No	`Azure`	
StartTime [UTC]	datetime	No	`2020-05-14 09:26:03.478039`	The time this alert was activated.
TenantId	string	No	`69ca04b0-703d-4b3a-9184-c4e3c15d6f5e`	Microsoft Entra tenant ID of the device.
TimeGenerated [UTC]	datetime	No	`2020-05-14 09:26:03.478039`	The time the snapshot generated this specific record. This is to determine to which batch snapshot this record belongs.
Type	string	No	`UCDeviceAlert`	The entity type
URL	string	No	`aka.ms/errordetail32152`	Currently, data isn't gathered to populate this field. An optional URL to get more in-depth information related to this alert.

Feedback

Was this page helpful? Yes No

Provide product feedback

UCDOAggregatedStatus

Article • 12/06/2023 • Applies to: ✅ Windows 11, ✅ Windows 10

UCDOAggregatedStatus is an aggregation of all individual UDDOStatus records across the tenant and summarizes bandwidth savings across all devices enrolled using Delivery Optimization and Microsoft Connected Cache.

Schema for UCDOAggregatedStatus

Expand table

Field	Type	Example	Description
AzureADDeviceId	string	`71db1a1a-f1a6-4a25-b88f-79c2f513dae0`	Microsoft Entra Device ID
AzureADTenantId	string	`69ca04b0-703d-4b3a-9184-c4e3c15d6f5e`	Microsoft Entra tenant ID
BWOptPercent28Days	real	`10.61`	Bandwidth optimization (as a percentage of savings of total bandwidth otherwise incurred) for this device. A rolling 28-day basis.
BytesFromCache	long	`285212672`	Total number of bytes that were delivered from Microsoft Connected Cache.
BytesFromCDN	long	`11463008693388`	Total number of bytes that were delivered from a Content Delivery Network (CDN).
BytesFromGroupPeers	long	`30830657175`	Total number of bytes that were delivered from Group peers, sharing the same GroupId.
BytesFromIntPeers	long	`285212672`	Total number of bytes that were delivered from Internet peers.
BytesFromPeers	long	`285212672`	Total number of bytes delivered via all peers.
ContentType	string	`Driver Updates`	One of the supported types of content.
DeviceCount	long	`27077`	Number of devices.

Field	Type	Example	Description
TenantId	string	`6yy5y416-2d35-3yyf-ab5f-aea713e489d2`	Tenant ID
TimeGenerated	datetime	`2022-11-17T22:11:40.1132971Z`	The time the snapshot generated this specific record. This is to determine to which batch snapshot this record belongs.
Type	string	`UCDOAggregatedStatus`	The entity type.

Feedback

Was this page helpful? Yes No

Provide product feedback

UCDOStatus

Applies to: ✅ Windows 11, ✅ Windows 10

UCDOStatus provides information, for a single device, on its bandwidth utilization across content types in the event they use Delivery Optimization and Microsoft Connected Cache.

Data schema for UCDOStatus

Expand table

Field	Type	Example	Description
AzureADDeviceId	string	`71db1a1a-f1a6-4a25-b88f-79c2f513dae0`	Microsoft Entra Device ID
AzureADTenantId	string	`69ca04b0-703d-4b3a-9184-c4e3c15d6f5e`	Microsoft Entra tenant ID
BWOptPercent28Days	real	`10.61`	Bandwidth optimization (as a percentage of savings of total bandwidth otherwise incurred) for this device. A rolling 28-day basis.
BWOptPercent7Days	real	`10.61`	Bandwidth optimization (as a percentage of savings of total bandwidth otherwise incurred) for this device. A rolling 7-day basis.

Field	Type	Example	Description
BytesFromCache	long	`285212672`	Total number of bytes that were delivered from Microsoft Connected Cache.
BytesFromCDN	long	`11463008693388`	Total number of bytes that were delivered from a Content Delivery Network (CDN).
BytesFromGroupPeers	long	`30830657175`	Total number of bytes that were delivered from Group peers, sharing the same GroupId.
BytesFromIntPeers	long	`285212672`	Total number of bytes that were delivered from Internet peers.
BytesFromPeers	long	`285212672`	Total number of bytes delivered via LAN peers.
City	string	`Redmond`	Approximate city where device was located while downloading content, based on IP address.

Field	Type	Example	Description
ContentDownloadMode	int	`1`	Device's Delivery Optimization Download Mode used to download content.
ContentType	string	`Driver Updates`	One of the supported types of content.
Country	string	`US`	Approximate country where device was located while downloading content, based on IP address.
DeviceName	string	`DESKTOP-DO`	User or organization provided device name. If the value appears as '#', configure the device to send device name.
DOStatusDescription	string	`Downloading`	A short description of Delivery Optimization status, if any.
DownloadMode	string	`LAN (1)`	Delivery Optimization Download Mode configured on the device.

Field	Type	Example	Description
DownloadModeSrc	string	`MDM`	The source of the Download Mode configuration.
GlobalDeviceId	string	`g:9832741921341`	Microsoft global device identifier. This identifier is used by Microsoft internally.
GroupID	string	`3suvw1efol0nmy8y9g8tfhtj1onwpsk9g9swpwnvfra=`	Delivery Optimization Group ID GUID value.
ISP	string	`Microsoft Corporation`	Internet Service Provider estimation.
LastCensusSeenTime	datetime	`2020-05-14 09:26:03.478039`	The last time this device performed a successful census scan, if any.
NoPeersCount	long	`4`	Count of peers device interacted with.
OSVersion	string	`1909`	The Windows 10/11 operating system version currently installed on the device, such as 20H1, 21H2.
PeerEligibleTransfers	long	`5`	Total count of eligible

Field	Type	Example	Description
			transfers by peers.
PeeringStatus	string	`On`	Delivery Optimization peering status.
PeersCannotConnectCount	long	`1`	Count of peers Delivery Optimization couldn't connect to.
PeersSuccessCount	long	`2`	Count of peers Delivery Optimization successfully connected to.
PeersUnknownCount	long	`0`	Count of peers with an unknown relation.
TenantId	string	`6yy5y416-2d35-3yyf-ab5f-aea713e489d2`	Tenant ID
TimeGenerated	datetime	`2022-11-17T22:11:40.1132971Z`	The time the snapshot generated this specific record. This is to determine to which batch snapshot this record belongs.
TotalTimeForDownload	string	`00:02:11`	Total time to download content.
TotalTransfers	long	`304`	Total count of data transfers needed to download content.

Field	Type	Example	Description
Type	string	`UCDOAggregatedStatus`	The entity type.

Last updated on 08/04/2025

UCServiceUpdateStatus

Article • 12/06/2023 • Applies to: ✅ Windows 11, ✅ Windows 10

Update Event that comes directly from the service-side. The event has only service-side information for one device (client), and one update, in one deployment. This event has certain fields removed from it in favor of being able to show data in near real time.

Schema for UCServiceUpdateStatus

Expand table

Field	Type	Enumerated type	Example	Description
AzureADDeviceId	string	No	`71db1a1a-f1a6-4a25-b88f-79c2f513dae0`	Microsoft Entra Device ID
AzureADTenantId	string	No	`69ca04b0-703d-4b3a-9184-c4e3c15d6f5e`	Microsoft Entra tenant ID
CatalogId	string	No	`b0f410599615e2ce15e6614ac3fc4ec62d80324020351e172edef89091a64f2f`	This field applies to drivers only. The Catalog ID of the update from Windows Autopatch.
DeploymentApprovedTime [UTC]	datetime	No	`2020-05-14 09:26:03.478039`	This field applies to drivers only. Date and time of the update approval
DeploymentId	string	No	`cf1b12a3-3d84-4ce3-bc8e-de48459e252d`	If this DeviceUpdateEvent is from content deployed by a deployment scheduler service policy, this GUID maps to that policy, otherwise it's empty.
DeploymentIsExpedited	bool	No	`1`	Currently, data isn't gathered to populate this field. It indicated whether the content is being expedited
DeploymentName	string	No	`My deployment`	Currently, data isn't gathered to populate this field. Friendly name of the created deployment
DeploymentRevokeTime [UTC]	datetime	No	`2020-05-14 09:26:03.478039`	This field applies to drivers only. Date and time the update was revoked
GlobalDeviceId	string	No	`g:9832741921341`	Currently, data isn't gathered to populate this field.

Field	Type	Enumerated type	Example	Description
				Microsoft internal global device identifier
OfferReadyTime [UTC]	datetime	No	`2020-05-14 09:26:03.478039`	DateTime of OfferReady transition. If empty, not yet been offered.
PolicyCreatedTime [UTC]	datetime	No	`2020-05-14 09:26:03.478039`	This field applies to drivers only. Date and time the policy was created
PolicyId	string	No	`9011c330-1234-5678-9abc-def012345678`	This field applies to drivers only. The policy identifier targeting the update to this device
PolicyName	string	No	`My policy`	Currently, data isn't gathered to populate this field. This field applies to drivers only. Friendly name of the policy.
ProjectedOfferReadyTime [UTC]	datetime	No	`2020-05-14 09:26:03.478039`	Projected time update will be offered to device. If empty, unknown.
ServiceState	string	Yes	`Offering`	High-level state of update's status relative to device, service-side.
ServiceSubstate	string	Yes	`OfferReady`	Low-level state of update's status relative to device, service-side.
ServiceSubstateRank	int	No		Currently, data isn't gathered to populate this field. Ranking of Substates for sequential ordering in funnel-type views. The rankings between ServiceSubstate and ClientSubstate can be used together.
ServiceSubstateTime [UTC]	datetime	No	`2020-05-14 09:26:03.478039`	Date and time of last ServiceSubstate transition.
SourceSystem	string	No	`Azure`	
TargetBuild	string	No	`10.0.18363.836`	The full build for the content this event is tracking.

Field	Type	Enumerated type	Example	Description
				For Windows 10, this string corresponds to "10.0.Build.Revision"
TargetVersion	string	No	`1909`	The version of content this DeviceUpdateEvent is tracking. For Windows 10 updates, this number would correspond to the year/month version format used, such as 1903.
TenantId	string	No	`9011c330-1234-5678-9abc-def012345678`	Microsoft Entra tenant ID
TimeGenerated [UTC]	datetime		`2020-05-14 09:26:03.478039`	Time the snapshot ran can also be the same as EventDateTimeUTC in some cases.
Type	string	No	`UCServiceUpdateStatus`	The entity type
UdpateIsSystemManifest	string	No		Currently, data isn't gathered to populate this field. This field applies to drivers only.
UpdateCategory	string	Yes	`WindowsFeatureUpdate`	The type of content this DeviceUpdateEvent is tracking.
UpdateClassification	string	Yes	`Upgrade`	Whether this update is an upgrade (feature update), security (quality update), nonsecurity (quality update), or driver
UpdateDisplayName	string	No	`Windows 10 1909`	The long-form display name for the given update. Varies on content type (feature update. quality update)
UpdateId	string	No	`10e519f0-06ae-4141-8f53-afee63e995f0`	This field applies to drivers only. Update ID of the targeted update
UpdateManufacturer	string	No	`Microsoft`	This field applies to drivers only. Manufacturer of update. Microsoft for feature or quality updates, for drivers the name of

Field	Type	Enumerated type	Example	Description
				driver manufacturer.
UpdateProvider	string	No	`Microsoft`	This field applies to drivers only. Update provider of drivers and firmware
UpdateRecommendedTime [UTC]	datetime	No	`2020-05-14 09:26:03.478039`	This field applies to drivers only. Date and time when the update was recommended to the device
UpdateReleaseTime [UTC]	datetime	No	`2020-05-14 09:26:03.478039`	Currently, data isn't gathered to populate this field. The release date of the update
UpdateVersion	string	No	`20.0.19.3`	This field applies to drivers only. Update version of drivers or firmware
UpdateVersionTime [UTC]	datetime	No	`2020-05-14 09:26:03.478039`	This field applies to drivers only. Update version date time stamp for drivers and firmware

Feedback

Was this page helpful? Yes No

Provide product feedback

UCUpdateAlert

Article • 12/06/2023 • Applies to: ✅ Windows 11, ✅ Windows 10

Alert for both client and service updates. Contains information that needs attention, relative to one device (client), one update, and one deployment (if relevant). Certain fields may be blank depending on the UpdateAlert's AlertType field; for example, ServiceUpdateAlert won't necessarily contain client-side statuses.

Schema for UCUpdateAlert

Expand table

Field	Type	ENUM	Example	Description
AlertClassification	string	Yes	`Error`	Whether this alert is an error, a warning, or informational
AlertData	string	No	`{ "freeDiskCapacityMb": 3213, "contentSizeMb": 4381}`	An optional string formatted as a json payload containing metadata for the alert.
AlertId	string	No	`9e107d9d372bb6826bd81d3542a419d6`	The unique identifier of this alert
AlertRank	int	No	`1000`	Integer ranking of alert for prioritization during troubleshooting
AlertStatus	string	Yes	`Active`	Whether this alert is active, resolved, or deleted
AlertSubtype	string	Yes	`DiskFull`	The subtype of alert
AlertType	string	Yes	`ClientUpdateAlert`	The type of alert such as ClientUpdateAlert or ServiceUpdateAlert. Indicates which fields are present.
AzureADDeviceId	string	No	`71db1a1a-f1a6-4a25-b88f-79c2f513dae0`	Microsoft Entra Device ID
AzureADTenantId	string	No	`69ca04b0-703d-4b3a-9184-c4e3c15d6f5e`	Microsoft Entra tenant ID
CatalogId	string	No	`b0f410599615e2ce15e6614ac3fc4ec62d80324020351e172edef89091a64f2f`	This field applies to drivers only. The Catalog ID of the update from Windows Autopatch.
ClientSubstate	string	Yes	`DownloadStart`	If the alert is from the client, the ClientSubstate at the time this alert was

Field	Type	ENUM	Example	Description
				activated or updated, else empty.
ClientSubstateRank	int	No	`2300`	Rank of ClientSubstate
DeploymentId	string	No	`cf1b12a3-3d84-4ce3-bc8e-de48459e252d`	The deployment this alert is relative to, if there's one.
Description	string	No	`Disk full`	A localized string translated from a combination of other Alert fields + language preference that describes the issue in detail.
DeviceName	string	No	`JohnPC-Contoso`	The given device's name
ErrorCode	string	No	`0x8326CFA2D_C3FD`	The error code, if any, that triggered this alert. In the case of client-based explicit alerts, error codes can have extended error codes, which are appended to the error code with an underscore separator.
ErrorSymName	string	No	`WU_E_DISK_FULL`	The symbolic name that maps to the error code, if any, otherwise empty.
GlobalDeviceId	string	No	`g:1298371934870`	Internal Microsoft Global identifier, if available.
Recommendation	string	No	`Free up disk space.`	A localized string translated from RecommendedAction, Message, and other fields (depending on the source of the alert) that provides a recommended action.
ResolvedTime [UTC]	datetime	No	`2020-05-14 09:26:03.478039`	The time this alert was resolved, else empty.
SCCMClientId	string	No	`5AB72FAC-93AB-4954-9AB0-6557D0EFA245`	Configuration manager client ID of the device, if available.
ServiceSubstate	string	Yes	`OfferReady`	If the alert is from the service, the ServiceSubstate at the time this alert was

Field	Type	ENUM	Example	Description
				activated or updated, else empty.
ServiceSubstateRank	int	No		Rank of 'ClientSubstate'
SourceSystem	string	No	`Azure`	
StartTime [UTC]	datetime	No	`2020-05-14 09:26:03.478039`	The time this alert was activated.
TargetBuild	string	No	`18363.836`	The Windows 10 Major. Revision this 'UpdateAlert' is relative to.
TargetVersion	string	No	`1909`	The Windows 10 build this UpdateAlert is relative to.
TenantId	string	No	`69ca04b0-703d-4b3a-9184-c4e3c15d6f5e`	Microsoft Entra tenant ID of the device.
TimeGenerated [UTC]	datetime	No	`2020-05-14 09:26:03.478039`	The time the snapshot generated this specific record. This is to determine to which batch snapshot this record belongs.
Type	string	No	`UCUpdateAlert`	The entity type
UpdateCategory	string	Yes	`WindowsFeatureUpdate`	The type of content this DeviceUpdateEvent is tracking.
UpdateClassification	string	Yes	`Upgrade`	Whether this update is an upgrade (feature update), security (quality update), nonsecurity (quality update), or driver
UpdateId	string	No	`10e519f0-06ae-4141-8f53-afee63e995f0`	This field applies to drivers only. The Update ID of the targeted update.
URL	string	No	`aka.ms/errordetail32152`	An optional URL to get more in-depth information related to this alert.

Feedback

Was this page helpful? Yes No

Provide product feedback

Enumerated types for Windows Update for Business reports

Article • 09/16/2024 • Applies to: ✅ Windows 11, ✅ Windows 10

The following enumerated types are used in Windows Update for Business reports:

OSEdition

SKU of Windows the device is running.

Expand table

Value	Description
Enterprise	Windows Enterprise
Professional	Windows Professional
ProfessionalWorkstation	Windows Professional workstation
ProfessionalN	Similar to Windows Professional edition but doesn't include Windows media player.
Education	Windows Education

OSArchitecture

Architecture of the OS running on the client.

Expand table

Value	Description
amd64	OS is 64-bit
x86	OS is 32-bit
Unknown	The OS architecture is unknown

OSFeatureUpdateStatus

Feature updates status

Expand table

Value	Description
Unknown	Default, sent if client data unavailable.
InService	Client is on a version of Windows 10 that is serviced.
EndOfService	Client is on a version of Windows 10 that is no longer serviced.

OSQualityUpdateStatus

Quality updates status

Expand table

Value	Description
Latest	Client is on the latest quality update
NotLatest	Client isn't on the latest quality update

OSSecurityUpdateStatus

Security updates status

Expand table

Value	Description
Latest	Client is on the latest security update
NotLatest	Client isn't on the latest security update
MultipleSecurityUpdatesMissing	Client is missing multiple security updates

OSFeatureUpdateComplianceStatus, OSSecurityUpdateComplianceStatus, OSQualityUpdateComplianceStatus

Compliance status

Expand table

Value	Description
Compliant	The latest deployment from Windows Autopatch is installed on the client
NotCompliant	The latest deployment from Windows Autopatch isn't installed on the client
NotApplicable	Client isn't part of any Windows Autopatch deployments

OSServicingChannel

Servicing channel of client

Expand table

Value	Description
Unknown	Default, release branch can't be defined.
SAC	Semi-annual release channel
LTSC	Long-term servicing channel
WIP-S	Windows Insider Preview - Slow ring
WIP-F	Windows Insider Preview - Fast ring
Internal	An identifiable, but internal release ring

ServiceState

High-level service state OSServicingChannel

Expand table

Value	Description
Pending	Windows Autopatch isn't targeting this update to this device because the update isn't ready.
Offering	Service is offering the update to the device. The update is available for the device to get if it scans Windows Update.
OnHold	Service is holding off on offering update to the device indefinitely. Until either the service or admin changes some condition, devices remain in this state.

Value	Description
Canceled	Service canceled offering update to the device, and the device is confirmed to not be installing the update.

ServiceSubstate

Lower-level service state

Expand table

Value	ServiceState
Validation	Update can't be offered to the device because a validation issue with the device and deployment service.
Scheduled	Update isn't ready to be offered to the device, but is scheduled for offering at a later date.
OfferReady	Update is currently being offered to the device from Windows Update.
RemovedFromDeployment	Update offering was canceled because it was removed from the deployment because of an explicit administrator action.
AdminCancelled	Update offering was canceled because of an explicit administrator action.
ServiceCancelled	Update offering was canceled because of an automatic action by the deployment service.
AdminPaused	Update is on hold because the deployment was paused with an explicit administrator action.
ServicePaused	Update is on hold because of an automatic action by the deployment service.
SafeguardHold	Update isn't offered because an existing safeguard hold on the device.

ClientState

High-level client state

Expand table

Value	Description
Unknown	Default value, if ClientSubstate is unknown (in other words, no client data)
Offering	Update is being offered to device
Installing	Update is in progress on device
Uninstalling	Update is being uninstalled from device
Installed	Update has been installed to device
Uninstalled	Update has been uninstalled from device
Canceled	Update has been canceled from device
OnHold	Update has been on Hold

ClientSubstate

Lower-level client state

Expand table

Value	Description
Unknown	Default value, if ClientSubstate is unknown (in other words, no client data)
Offering	Update is being offered to device
Installing	Update is in progress on device
Uninstalling	Update is being uninstalled from device
Installed	Update has been installed to device
Uninstalled	Update has been uninstalled from device
Canceled	Update has been canceled from device
OnHold	Update has been on Hold

UpdateCategory

Type of update.

Expand table

Value	Description
WindowsQualityUpdate	Windows feature update
WindowsFeatureUpdate	Windows quality update
DriverUpdate	Driver update

UpdateClassification

Whether this update is an upgrade, security, nonsecurity, or driver

Expand table

Value	Description
Security	Update is a quality update containing security fixes
NonSecurity	Update is a quality update not containing security fixes
Upgrade	Update is a feature update

UpdateSource

Source of the update

Expand table

Value	Description
Inferred	
MuV6	Update through old Windows Update, or via WSUS (uses old protocol)
UUP	Update through modern Windows Update

ReadinessStatus

Whether the device is capable of taking target OS and version.

Expand table

Value	Description
Capable	The device meets all requirements to upgrade to Windows 11.
Not Capable	The device doesn't meet the requirements to upgrade to Windows 11. Check Readiness Reason for the reason.
Unknown	Microsoft doesn't have enough data points to determine the eligibility status.

ReadinessReason

Reason why the device isn't capable of updating to target OS and version.

Expand table

Value	Description
tpm	Trusted Platform Module (TPM) version 2.0 is required. If your device doesn't meet the minimum requirements because of TPM, see Enable TPM 2.0 on your PC to see if there are any remediation steps you can take.
cpufms	CPU not supported. For more information, see Windows Processor Requirements
sysdrivesize	64 GB or larger storage device required. If your PC doesn't have a large enough storage drive, sometimes there are options for upgrading the drive. Consult your PC manufacturer's website or with a retailer to see if there are options to meet the minimum requirements for Windows 11.
UefiSecureBoot	UEFI (Unified Extensible Firmware Interface) and Secure Boot capability. If your device doesn't meet the minimum requirements because it's not Secure Boot capable. For more information, see Windows 11 and Secure Boot to see if you're able to enable Secure Boot. Secure Boot can only be enabled with UEFI.

AlertType

Type of alert.

Expand table

Value	Description
ServiceUpdateAlert	Alert is relevant to Windows Autopatch's offering of the content to the client.

Value	Description
ClientUpdateAlert	Alert is relevant to client's ability to progress through the installation of the update content.
ServiceDeviceAlert	Alert is relevant to device's status within Windows Autopatch
ClientDeviceAlert	Alert is relevant to device's state
DeploymentAlert	Alert is relevant to an entire deployment, or a significant number of devices in the deployment.

AlertSubtype

Subtype of alert.

Expand table

Value	Description
CancelledByUser	The user canceled the update.
CertificateIssue	An expired certificate was encountered.
DamagedMedia	The update file appears to be damaged.
DeviceRegistrationInvalidAzureADJoin	Device isn't able to register or authenticate properly with the deployment service due to not being device-level Entra ID joined. Devices that are workplace-joined aren't compatible with the deployment service.
DiskFull	An operation couldn't be completed because the disk is full.
DiskIssue	Windows Update has found disk corruption.
DownloadCancelled	The download was canceled.
DownloadCredentialsIssue	A proxy server or firewall on your network might require credentials.
DownloadIssue	There was a download issue.
DownloadIssueServiceDisabled	The service the download depends on is disabled.
DownloadTimeout	A timeout occurred.
EndOfService	Client OS is no longer being serviced

Value	Description
EndOfServiceApproaching	Client OS servicing period completes in less than 60 days
FileNotFound	The installer couldn't find a Windows component that it needs.
InstallAccessDenied	Access denied.
InstallCancelled	Install canceled.
InstallFileLocked	Couldn't access the file because it's already in use.
InstallIssue	There was an installation issue.
InstallSetupBlock	There's an application or driver blocking the upgrade.
InstallSetupError	Encountered an error while installing the new version of Windows.
InstallSetupRestartRequired	A restart is required.
InstallSharingViolation	An application is likely interfering with Windows Update.
InstallSystemError	A system error occurred while installing the new version of Windows.
InsufficientUpdateConnectivity	Device hasn't had sufficient connectivity to Windows Update to progress through the update process and will experience delays.
MultipleSecurityUpdatesMissing	Client is missing multiple security updates
NetworkIssue	The server timed out waiting for the requested.
PathNotFound	The specified path can't be found.
RestartIssue	The restart to apply updates is being blocked by one or more applications.
SafeguardHold	Update can't be installed due to a known Safeguard Hold.
UnexpectedShutdown	The installation stopped because Windows was shutting down or restarting.
WindowsComponentCorruption	This device has a corrupted Windows component
WUBusy	Windows Update tried to install an update while another installation process was already running.

Value	Description
WUComponentMissing	Windows Update might be missing a component or the update file might be damaged.
WUDamaged	The update file might be damaged.
WUFileCorruption	Windows Update encountered corrupted files.
WUIssue	An unexpected issue was encountered during the installation.
WUSetupError	The setup process was suspended.

AlertStatus

Status of alert

Expand table

Value	Description
Active	Alert is active, still requires attention.
Resolved	Alert is resolved and no longer requires attention.
Deleted	Alert was deleted from the backend system.

AlertClassification

Whether this alert is an error, a warning, or informational.

Expand table

Value	Description
Informational	Alert is informational in nature.
Warning	Alert is a warning
Error	Alert is an error, or is related to an error. There should be an error code that maps to either something from the client or from the service.
Recommendation	Alert is a recommendation, something to optimize.

Feedback

Was this page helpful? Yes No

Provide product feedback

Resolve Windows upgrade errors: Technical information for IT Pros

Applies to: ✅ Windows 11, ✅ Windows 10

> ⓘ **Important**
>
> This article contains technical instructions for IT administrators. The article isn't intended for non-IT administrators such as home or consumer users.

This article contains a brief introduction to the Windows installation processes, and provides resolution procedures that IT administrators can use to resolve issues with a Windows upgrade.

The article is divided into subtopics of different technical levels. Basic level provides common procedures that can resolve several types of upgrade errors. Advanced level requires some experience with detailed troubleshooting methods.

The following four levels are assigned:

- Level 100: Basic
- Level 200: Moderate
- Level 300: Moderate advanced
- Level 400: Advanced

In this guide

See the following articles in this section:

- Quick fixes: (Level 100) Steps to take to eliminate many Windows upgrade errors.
- SetupDiag: (Level 300) SetupDiag is a new tool to help isolate the root cause of an upgrade failure.
- Troubleshooting upgrade errors: (Level 300) General advice and techniques for troubleshooting Windows upgrade errors, and an explanation of phases used during the upgrade process.
- Windows Error Reporting: (Level 300) How to use Event Viewer to review details about a Windows upgrade.
- Upgrade error codes: (Level 400) The components of an error code are explained.
  - Result codes: Information about result codes.
  - Extend codes: Information about extend codes.
- Log files: (Level 400) A list and description of log files useful for troubleshooting.
  - Log entry structure: The format of a log entry is described.

  - Analyze log files: General procedures for log file analysis, and an example.
- Resolution procedures: (Level 200) Causes and mitigation procedures associated with specific error codes.
  - 0xC1900101: Information about the 0xC1900101 result code.
  - 0x800xxxxx: Information about result codes that start with 0x800.
  - Other result codes: Additional causes and mitigation procedures are provided for some result codes.
  - Other error codes: Additional causes and mitigation procedures are provided for some error codes.
- Submit Windows upgrade errors: (Level 100) Submit upgrade errors to Microsoft for analysis.

Related articles

- Fix Windows Update errors by using the DISM or System Update Readiness tool.

Last updated on 04/08/2025

Windows 10 upgrade quick fixes

01/15/2025Applies to: Supported versions of Windows Client

Applies to: Windows 10

> **Note**
>
> This is a 100 level topic (basic).
>
> For IT professionals, check more information in Resolve Windows 10 upgrade errors.

The following list of fixes can resolve many Windows upgrade problems. You should try these steps before contacting Microsoft support, or attempting a more advanced analysis of a Windows upgrade failure. Also review information at Windows 10 help.

> **Tip**
>
> You might also wish to try a new tool available from Microsoft that helps to diagnose many Windows upgrade errors. For more information and to download this tool, see SetupDiag. The topic is more advanced (300 level) because several advanced options are available for using the tool. However, you can now just download and then double-click the tool to run it. By default when you click Save, the tool is saved in your *Downloads* folder. Double-click the tool in the folder and wait until it finishes running (it might take a few minutes), then double-click the *SetupDiagResults.log* file and open it using Notepad to see the results of the analysis.

List of fixes

Here are the step-by-step instructions:

1. Remove nonessential external hardware, such as docks and USB devices.
2. Check the system drive for errors and attempt repairs.
3. Run the Windows Update troubleshooter.
4. Attempt to restore and repair system files.
5. Update Windows so that all available recommended updates are installed, and ensure the computer is rebooted if it is necessary to complete installation of an update.
6. Temporarily uninstall non-Microsoft antivirus software.
7. Uninstall all nonessential software.
8. Update firmware and drivers.

9. Ensure that "Download and install updates (recommended)" is accepted at the start of the upgrade process.
10. Verify at least 16 GB of free space is available to upgrade a 32-bit OS, or 20 GB for a 64-bit OS.

Remove external hardware

If the computer is portable and it's currently in a docking station, undock the computer.

Unplug nonessential external hardware devices from the computer, such as:

- Headphones
- Joysticks
- Printers
- Plotters
- Projectors
- Scanners
- Speakers
- USB flash drives
- Portable hard drives
- Portable CD/DVD/Blu-ray drives
- Microphones
- Media card readers
- Cameras/Webcams
- Smart phones
- Secondary monitors, keyboards, mice

For more information about disconnecting external devices, see Safely remove hardware in Windows 10

Repair the system drive

The system drive is the drive that contains the system partition. It is usually the C: drive.

To check and repair errors on the system drive:

1. Select **Start**.

2. Type *command*.

3. Right-click **Command Prompt** and then select **Run as administrator**.

4. If you're prompted by UAC, select **Yes**.

5. Type `chkdsk /F` and press Enter.

6. When you're prompted to schedule a check the next time the system restarts, type `Y`.

7. See the following example.

Console

```
C:\WINDOWS\system32>chkdsk /F
The type of the file system is NTFS.
Cannot lock current drive.

Chkdsk cannot run because the volume is in use by another
process.  Would you like to schedule this volume to be
checked the next time the system restarts? (Y/N) Y

This volume will be checked the next time the system restarts.
```

8. Restart the computer. The computer will pause before loading Windows and perform a repair of your hard drive.

Windows Update Troubleshooter

The Windows Update Troubleshooter tool will automatically analyze and fix problems with Windows Update, such as a corrupted download. It will also tell you if there's a pending reboot that is preventing Windows from updating.

Download the tool for Windows 10.

To run the tool, select the appropriate link above. Your web browser will prompt you to save or open the file. Select **open** and the tool will automatically start. The tool will walk you through analyzing and fixing some common problems.

Repair system files

This fix is also described in detail at answers.microsoft.com.

To check and repair system files:

1. Select **Start**.

2. Type *command*.

3. Right-click **Command Prompt** and then select **Run as administrator**.

4. If you're prompted by UAC, select **Yes**.

5. Type `sfc /scannow` and press Enter. See the following example:

```
Console

C:\>sfc /scannow

Beginning system scan.  This process will take some time.

Beginning verification phase of system scan.
Verification 100% complete.

Windows Resource Protection did not find any integrity violations.
```

6. If you're running Windows 8.1 or later, type `DISM.exe /Online /Cleanup-image /Restorehealth` and press Enter (the `DISM` command options aren't available for Windows 7). See the following example:

```
Console

C:\>DISM.exe /Online /Cleanup-image /Restorehealth

Deployment Image Servicing and Management tool
Version: 10.0.16299.15

Image Version: 10.0.16299.309

[==========================100.0%==========================] The restore
operation completed successfully.
The operation completed successfully.
```

> **Note**
>
> It may take several minutes for the command operations to be completed. For more information, see Repair a Windows Image and Use the System File Checker tool.

Update Windows

You should ensure that all important updates are installed before attempting to upgrade. This includes updates to hardware drivers on your computer.

Answer questions that the agent asks, and follow instructions to ensure that Windows is up to date. You can also run the Windows Update Troubleshooter described above.

Select **Start**, select **Power Options**, and then restart the computer.

Uninstall non-Microsoft antivirus software

Use Windows Defender for protection during the upgrade.

Verify compatibility information, and if desired reinstall antivirus applications after the upgrade. If you plan to reinstall the application after upgrading, be sure that you have the installation media and all required activation information before removing the program.

To remove the application, go to **Control Panel** > **Programs** > **Programs and Features** and select the antivirus application, then select **Uninstall**. Choose **Yes** when you're asked to confirm program removal.

For more information, see Windows 7 - How to properly uninstall programs or Repair or remove programs in Windows 10.

Uninstall non-essential software

Outdated applications can cause problems with a Windows upgrade. Removing old or non-essential applications from the computer can therefore help.

If you plan to reinstall the application later, be sure that you have the installation media and all required activation information before removing it.

To remove programs, use the same steps as are provided above for uninstalling non-Microsoft antivirus software, but instead of removing the antivirus application repeat the steps for all your non-essential, unused, or out-of-date software.

Update firmware and drivers

Updating firmware (such as the BIOS) and installing hardware drivers is an advanced task. Don't attempt to update BIOS if you aren't familiar with BIOS settings or aren't sure how to restore the previous BIOS version if there are problems. Most BIOS updates are provided as a "flash" update. Your manufacturer might provide a tool to perform the update, or you might be required to enter the BIOS and update it manually. Be sure to save your working BIOS settings, since some updates can reset your configuration and make the computer fail to boot if (for example) a RAID configuration is changed.

Most BIOS and other hardware updates can be obtained from a website maintained by your computer manufacturer. For example, Microsoft Surface device drivers can be obtained at: Download the latest firmware and drivers for Surface devices.

To obtain the proper firmware drivers, search for the most updated driver version provided by your computer manufacturer. Install these updates and reboot the computer after installation.

Request assistance from the manufacturer if you have any questions.

Ensure that "Download and install updates" is selected

When you begin a Windows Update, the setup process will ask you to **Get important updates**. Answer **Yes** if the computer you're updating is connected to the Internet. See the following example:

Verify disk space

You can see a list of requirements for Windows 10 at Windows 10 Specifications & System Requirements. One of the requirements is that enough hard drive space be available for the installation to take place. At least 16 GB of free space must be available on the system drive to upgrade a 32-bit OS, or 20 GB for a 64-bit OS.

To view how much hard drive space is available on your computer, open File Explorer. In Windows 7, this was called Windows Explorer.

In **File Explorer**, select **Computer** or **This PC** on the left, then look under **Hard Disk Drives** or under **Devices and drives**. If there are multiple drives listed, the system drive is the drive that

includes a Microsoft Windows logo above the drive icon.

The amount of space available on the system drive will be displayed under the drive. See the following example:

In the previous example, there's 703 GB of available free space on the system drive (C:).

To free up more space on the system drive, begin by running Disk Cleanup. You can access Disk Cleanup by right-clicking the hard drive icon and then clicking Properties. See the following example:

For instructions to run Disk Cleanup and other suggestions to free up hard drive space, see Tips to free up drive space on your PC.

When you run Disk Cleanup and enable the option to Clean up system files, you can remove previous Windows installations, which can free a large amount of space. You should only do this if you don't plan to restore the old OS version.

Open an elevated command prompt

Tip

It is no longer necessary to open an elevated command prompt to run the SetupDiag tool. However, this is still the optimal way to run the tool.

To launch an elevated command prompt, press the Windows key on your keyboard, type *cmd*, press Ctrl+Shift+Enter, and then select **Yes** to confirm the elevation prompt. For more information about screenshots and other steps to open an elevated command prompt, see Command Prompt (Admin) Windows 7.

> **Note**
>
> When you open an elevated command prompt, you will usually start in the *C:\WINDOWS\system32* directory. To run a program that you recently downloaded, you must change to the directory where the program is located. Alternatively, you can move or copy the program to a directory in your PATH variable. These directories are automatically searched. Type `echo %PATH%` to see the directories in your PATH variable.

Another option is to use File Explorer to create a new folder under C: with a short name such as "new" then copy or move the programs you want to run (like SetupDiag) to this folder using File Explorer. When you open an elevated command prompt, change to this directory by typing `cd c:\new`, and now you can run the programs in that folder.

If you downloaded the *SetupDiag.exe* program to your computer, then copied it to the folder *C:\new*, and you opened an elevated command prompt then typed `cd c:\new` to change to this directory, you can just type `setupdiag` and press Enter to run the program. This program will analyze the files on your computer to see why a Windows Upgrade failed and if the reason was a common one, it will report this reason. It will not fix the problem for you but knowing why the upgrade failed enables you to take steps to fix the problem.

Data collection

If you need assistance from Microsoft support, we recommend you collect the information by following the steps mentioned in Gather information by using TSS for deployment-related issues.

Reference

- Windows 10 FAQ for IT professionals
- Windows 10 Enterprise system requirements
- Windows 10 Specifications

- Windows 10 IT pro forums
- Fix Windows Update errors by using the DISM or System Update Readiness tool

SetupDiag

Applies to: ✅ Windows 11, ✅ Windows 10

> **Note**
>
> This article is a 300 level article (moderate advanced). See Resolve Windows upgrade errors for a full list of articles in this section.

About SetupDiag

> **Important**
>
> When SetupDiag is run manually, Microsoft recommends running the latest version of SetupDiag. The latest version is available via the following link:
>
> Download the latest version of SetupDiag
>
> Running the latest version ensures the latest functionality and fixes known issues.

SetupDiag is a diagnostic tool that can be used to obtain details about why a Windows upgrade was unsuccessful.

SetupDiag works by examining Windows Setup log files. It attempts to parse these log files to determine the root cause of a failure to update or upgrade the computer to Windows. SetupDiag can be run on the computer that failed to update. The logs can also be exported from the computer to another location and then running SetupDiag in offline mode.

SetupDiag is included with Windows Setup in all currently supported versions of Windows.

During the upgrade process, Windows Setup extracts all its source files, including `SetupDiag.exe`, to the `%SystemDrive%\$Windows.~bt\Sources` directory. If there's an issue with the upgrade, SetupDiag automatically runs to determine the cause of the failure.

When run by Windows Setup, the following parameters are used:

- `/ZipLogs:False`
- `/Format:xml`
- `/Output:%windir%\logs\SetupDiag\SetupDiagResults.xml`
- `/RegPath:HKEY_LOCAL_MACHINE\SYSTEM\Setup\SetupDiag\Results`

The resulting SetupDiag analysis can be found at `%WinDir%\Logs\SetupDiag\SetupDiagResults.xml` and in the registry under `HKLM\SYSTEM\Setup\SetupDiag\Results`.

> **Note**
>
> When Windows Setup runs SetupDiag automatically, the registry path isn't the same as the default registry path when SetupDiag is run manually. When SetupDiag is run manually, and the `/RegPath` parameter isn't specified, data is stored in the registry at `HKLM\SYSTEM\Setup\MoSetup\Volatile\SetupDiag`.

> **Important**
>
> When SetupDiag indicates that there were multiple failures, the last failure in the log file is typically the fatal error, not the first one.

If the upgrade process proceeds normally, the `Sources` directory including `SetupDiag.exe` is moved under `%SystemDrive%\Windows.Old` for cleanup. If the `Windows.old` directory is deleted later, `SetupDiag.exe` is also removed.

> **Tip**
>
> If `SetupDiag.exe` is needed after the `Windows.old` directory is deleted, it can be downloaded from the Microsoft Download Center.

Requirements

1. The destination version of Windows must be a currently supported version of Windows. The originally installed version of Windows can be a version of Windows that's out of support as long as:

    - The destination version of Windows is a currently supported version of Windows.
    - Upgrade to the destination version of Windows is supported from the original installed version of Windows.

2. .NET Framework 4.7.2 or newer must be installed. To determine which version of .NET is preinstalled with a specific version of Windows, see .NET Framework system requirements: Supported client operating systems. To determine which version of .NET is currently installed, see How to: Determine Which .NET Framework Versions Are Installed.

The following command-line query can be used to display the currently installed version of .NET:

Windows Command Prompt

```
reg.exe query "HKLM\SOFTWARE\Microsoft\Net Framework Setup\NDP\v4" /s
```

As long as at least the required version of .NET is installed, no additional action is required, including if a newer version is installed.

3. **Setupdiag.exe** should be run from an elevated command prompt for it to work properly.

Using SetupDiag

To use SetupDiag:

1. Verify that the system meets the requirements.
2. Download the latest version of SetupDiag.
3. If the web browser asks what to do with the file, choose **Save**. By default, the file is saved to the **Downloads** folder, which is displayed in File Explorer under **Quick access** in the left navigation pane. If desired, the file can also be saved to a different location by using **Save As**.
4. Once SetupDiag finishes downloading, open an elevated command prompt and navigate to the directory where `setupdiag.exe` was downloaded and saved to.
5. In the elevated command prompt, run `setupdiage.exe` in online mode using the desired parameters as documented in the Parameters and Examples sections.
6. Wait for SetupDiag to finish.
7. When SetupDiag finishes, two files are created in the same folder where SetupDiag was run from:
   - A configuration file.
   - A log file.
8. Use Notepad to open the log file **SetupDiagResults.log**.
9. Review the information that is displayed. If a rule was matched, this information can say why the computer failed to upgrade, and potentially how to fix the problem. See the section Text log sample.

> **Tip**
>
> For instructions on how to run the tool in offline mode in Windows PE, see the sections Parameters and Examples.

Parameters

Expand table

Parameter	Description
/?	Displays help information
/Output:[Full path and file name for output log file]	This optional parameter specifies the name and location for the results log file. The output file contains the analysis from SetupDiag. Only text format output is supported. UNC paths work provided the context under which SetupDiag runs has access to the UNC path. If the path has a space in it, the entire path must be enclosed in double quotes ("). See the Examples sections for an example. Default: If not specified, SetupDiag creates the file **SetupDiagResults.log** in the same directory where **SetupDiag.exe** is run.
/LogsPath:[Full path to logs]	This optional parameter specifies the location of logs to parse and where to find the log files for an offline analysis. These log files can be in a flat folder format, or containing multiple subdirectories. SetupDiag recursively searches all child directories. Defaults to checking the current system for logs.
/ZipLogs:[True | False]	This optional parameter Tells **SetupDiag.exe** to create a zip file containing the results and all the log files that were parsed. The zip file is created in the same directory where **SetupDiag.exe** is run. Default: If not specified, a value of 'true' is used.
/Format:[xml | json]	This optional parameter specifies the output format for log files to be XML or JSON. If this parameter isn't specified, text format is used by default.
/Scenario:[Recovery | Debug]	This optional parameter can do one of the following two items based on the argument used: • Recovery instructs **SetupDiag.exe** to look for and process reset and recovery logs and ignore setup/upgrade logs. • Debug instructs **SetupDiag.exe** to debug memory dumps if the requisite debug binaries are installed.
/Verbose	This optional parameter creates a diagnostic log in the current directory, with debugging information, additional data, and details about SetupDiag. By default, SetupDiag only

Parameter	Description
	produces a log file entry for major errors. Using **/Verbose** causes SetupDiag to always produce another log file with debugging details. These details can be useful when reporting a problem with SetupDiag.
/NoTel	This optional parameter tells **SetupDiag.exe** not to send diagnostic telemetry to Microsoft.
/RegPath	This optional parameter Instructs **SetupDiag.exe** to add failure information to the registry under the given path. Registry paths should start with **HKEY_LOCAL_MACHINE** or **HKEY_CURRENT_USER** and be accessible at the elevation level SetupDiag is executed under. If this parameter isn't specified, the default path is `HKLM\SYSTEM\Setup\MoSetup\Volatile\SetupDiag`.
/AddReg	This optional parameter Instructs **SetupDiag.exe** to add failure information to the registry on the executing system in offline mode. SetupDiag by default adds failure information to the registry in Online mode only. Registry data goes to `HKEY_LOCAL_MACHINE\SYSTEM\Setup\MoSetup\Volatile\SetupDiag` unless otherwise specified.

> **Note**
>
> The **/Mode** parameter is deprecated in SetupDiag.
>
> In previous versions, the **/Mode** parameter was used with the **/LogsPath** parameter in offline mode and would analyze a set of log files that were captured on a different computer. In current versions of SetupDiag, when **/LogsPath** is specified, then SetupDiag automatically runs in offline mode, therefore the **/Mode** parameter isn't needed.

Examples

> **Important**
>
> **SetupDiage.exe** should be run from an elevated command prompt for it to work properly.

- In the following example, SetupDiag is run without parameters and it displays help information.

Windows Command Prompt

```
SetupDiag.exe
```

- In the following example, SetupDiag is run in online mode (this mode is the default). It knows where to look for logs on the current (failing) system, so there's no need to gather logs ahead of time. A custom location for results is specified.

  Windows Command Prompt

  ```
  SetupDiag.exe /Output:C:\SetupDiag\Results.log
  ```

- The following example uses the **/Output** parameter to save results to a path name that contains a space:

  Windows Command Prompt

  ```
  SetupDiag /Output:"C:\Tools\SetupDiag\SetupDiag Results\Results.log"
  ```

- The following example specifies that SetupDiag is to run in offline mode, and to process the log files found in **D:\Temp\Logs\LogSet1**.

  Windows Command Prompt

  ```
  SetupDiag.exe /Output:C:\SetupDiag\Results.log /LogsPath:D:\Temp\Logs\LogSet1
  ```

- The following is an example of Reset/Recovery Offline Mode. SetupDiag is instructed to look for reset/recovery logs in the specified LogsPath location. It then outputs the results to the directory specified by the **/Output** parameter.

  Windows Command Prompt

  ```
  SetupDiag.exe /Output:C:\SetupDiag\RecoveryResults.log
  /LogsPath:D:\Temp\Cabs\PBR_Log /Scenario:Recovery
  ```

- The following example is an example of Reset/Recovery Online Mode. SetupDiag is instructed to look for reset/recovery logs on the current system and output its results in XML format.

  Windows Command Prompt

  ```
  SetupDiag.exe /Scenario:Recovery /Format:xml
  ```

- The following example is an example of Offline Mode. SetupDiag is instructed to parse setup/upgrade log files in the LogsPath directory and output the results to `C:\SetupDiag\Results.txt`.

Windows Command Prompt

```
SetupDiag.exe /Output:C:\SetupDiag\Results.txt /LogsPath:D:\Temp\Logs\Logs1
/RegPath:HKEY_CURRENT_USER\SYSTEM\SetupDiag
```

- The following example is an example of Online Mode. SetupDiag is instructed to look for setup/upgrade logs on the current system and output its results in XML format to `C:\SetupDiag\Results.xml`.

Windows Command Prompt

```
SetupDiag.exe /Output:C:\SetupDiag\Results.xml /Format:xml
```

Log files

Windows Setup Log Files and Event Logs has information about where logs are created during Windows Setup. For offline processing, SetupDiag should be run against the contents of the entire folder. For example, depending on when the upgrade failed, copy one of the following folders to the offline location:

- `\$Windows.~bt\sources\panther`
- `\$Windows.~bt\Sources\Rollback`
- `\Windows\Panther`
- `\Windows\Panther\NewOS`

If the parent folder and all subfolders are copied, SetupDiag automatically searches for log files in all subdirectories.

Setup bug check analysis

When Microsoft Windows encounters a condition that compromises safe system operation, the system halts. This condition is called a bug check. This condition is also commonly referred to as a system crash, a kernel error, a Stop error, or BSOD. Typically a hardware device, hardware driver, or related software causes this error.

If crash dumps are enabled on the system, a crash dump file is created. If the bug check occurs during an upgrade, Windows Setup extracts a minidump (`setupmem.dmp`) file. SetupDiag can also debug these setup-related minidumps.

To debug a setup-related bug check:

- Specify the **/LogsPath** parameter. Memory dumps can't be debugged in online mode.

- Gather the setup memory dump file (`setupmem.dmp) from the failing system.

  `Setupmem.dmp` is created in either **%SystemDrive%$Windows.~bt\Sources\Rollback**, or in **%WinDir%\Panther\NewOS\Rollback** depending on when the bug check occurs.

- Install the Windows Debugging Tools on the computer that runs SetupDiag.

In the following example, the `setupmem.dmp` file is copied to the `D:\Dump` directory and the Windows Debugging Tools are installed before running SetupDiag:

Windows Command Prompt

```
SetupDiag.exe /Output:C:\SetupDiag\Dumpdebug.log /LogsPath:D:\Dump
```

Known issues

- Some rules can take a long time to process if the log files involved are large.

Sample output

The following command is an example where SetupDiag is run in offline mode.

Windows Command Prompt

```
D:\SetupDiag>SetupDiag.exe /output:c:\setupdiag\result.xml
/logspath:D:\Tests\Logs\f55be736-beed-4b9b-aedf-c133536c946e /format:xml

SetupDiag v1.7.0.0
Copyright (c) Microsoft Corporation. All rights reserved.

Searching for setup logs...
Found d:\tests\Logs\f55be736-beed-4b9b-aedf-c133536c946e\setupact_6.log with update
date 6/12/2023 2:44:20 PM to be the correct setup log.
Found d:\tests\Logs\f55be736-beed-4b9b-aedf-c133536c946e\setupact_1.log with update
date 6/12/2023 2:45:19 PM to be the correct rollback log.

Gathering baseline information from setup logs...

SetupDiag: processing rule: CompatScanOnly.
...No match.

...

SetupDiag: processing rule: DISMImageSessionFailure.
..
Error: SetupDiag reports DISM provider failure.
Last Phase: Safe OS
Last Operation: Apply Optional Component status
```

```
Message = Failed to get the IDismImage instance from the image session
Function: CDISMManager::CloseImageSession
Error: 0x800706ba
Recommend you re-download the update source files, reboot and try the update again.

SetupDiag found 1 matching issue.

SetupDiag results were logged to: c:\setupdiag\results.xml
Logs ZipFile created at: c:\setupdiag\Logs_14.zip
```

Rules

When SetupDiag searches log files, it uses a set of rules to match known issues. These rules are contained in an xml file. The xml file might be updated with new and updated rules as new versions of SetupDiag are made available.

Each rule name and its associated unique rule identifier are listed with a description of the known upgrade-blocking issue. In the rule descriptions, the term **downlevel** refers to the first phase of the upgrade process, which runs under the original OS.

Expand table

Rule Name	GUID	Description
CompatScanOnly	FFDAFD37-DB75-498A-A893-472D49A1311D	This rule indicates that `setup.exe` was called with a specific command line parameter that indicated setup was to do a compatibility scan only, not an upgrade.
PlugInComplianceBlock	D912150B-1302-4860-91B5-527907D08960	Detects all compatibility blocks from Server compliance plug-ins. This rule is for server upgrades only. It outputs the compliance block and remediation required.
BitLockerHardblock	C30152E2-938E-44B8-915B-D1181BA635AE	This block is an upgrade block when the target OS doesn't support BitLocker, yet the host OS has BitLocker enabled.
VHDHardblock	D9ED1B82-4ED8-4DFD-8EC0-BE69048978CC	This block happens when the host OS is booted to a VHD image. Upgrade isn't supported when the host OS is booted from a VHD image.

Rule Name	GUID	Description
PortableWorkspaceHardblock	5B0D3AB4-212A-4CE4-BDB9-37CA404BB280	This block indicates that the host OS is booted from a Windows To-Go device (USB key. Upgrade isn't supported in the Windows To-Go environment.
AuditModeHardblock	A03BD71B-487B-4ACA-83A0-735B0F3F1A90	This block indicates that the host OS is currently booted into Audit Mode, a special mode for modifying the Windows state. Upgrade isn't supported from this state.
SafeModeHardblock	404D9523-B7A8-4203-90AF-5FBB05B6579B	This block indicates that the host OS is booted to Safe Mode, where upgrade isn't supported.
InsufficientSystemPartitionDiskSpaceHardblock	3789FBF8-E177-437D-B1E3-D38B4C4269D1	This block is encountered when setup determines the system partition doesn't have enough space to be serviced with the newer boot files required during the upgrade process. The system partition is where the boot loader files are stored
CompatBlockedApplicationAutoUninstall	BEBA5BC6-6150-413E-8ACE-5E1EC8D34DD5	This rule indicates there's an application that needs to be uninstalled before setup can continue.
CompatBlockedApplicationDismissable	EA52620B-E6A0-4BBC-882E-0686605736D9	When setup is run in **/quiet** mode, there are dismissible application messages that turn into blocks unless the command line also specifies **/compat ignorewarning**. This rule indicates setup was executed in **/quiet** mode but there's an application dismissible block message that prevented setup from continuing.
CompatBlockedFODDismissable	7B693C42-793E-4E9E-A10B-ED0F33D45E2A	When setup is run in **/quiet** mode, there are dismissible Feature On Demand messages that turn into blocks unless the

Rule Name	GUID	Description
		command line also specifies **/compat ignorewarning**. This rule indicates setup was executed in **/quiet** mode but there's a Feature On Demand dismissible block message that prevented setup from continuing, usually that the target OS image is missing a Feature On Demand that is installed in the current OS. Removal of the Feature On Demand in the current OS should also resolve the issue.
CompatBlockedApplicationManualUninstall	9E912E5F-25A5-4FC0-BEC1-CA0EA5432FF4	This rule indicates that an application without an Add/Remove Programs entry, is present on the system and blocking setup from continuing. This block typically requires manual removal of the files associated with this application to continue.
GenericCompatBlock	511B9D95-C945-4F9B-BD63-98F1465E1CF6	The rule indicates that system doesn't meet a hardware requirement for running Windows. For example, the device is missing a requirement for TPM 2.0. This issue can occur even when an attempt is made to bypass the hardware requirements.
GatedCompatBlock	34A9F145-3842-4A68-987F-4622EE0FC162	This rule indicates that the upgrade failed due to a temporary block. A temporary block is put in place when an issue is found with a specific piece of software or hardware driver and the issue has a fix pending. The block is lifted once the fix is widely available.
HardblockDeviceOrDriver	ED3AEFA1-F3E2-4F33-8A21-184ADF215B1B	This error indicates a device driver that is loaded on the host OS isn't compatible with the newer OS version. The device

Rule Name	GUID	Description
		driver needs to be removed before the upgrade.
HardblockMismatchedLanguage	60BA8449-CF23-4D92-A108-D6FCEFB95B45	This rule indicates the host OS and the target OS language editions don't match.
HardblockFlightSigning	598F2802-3E7F-4697-BD18-7A6371C8B2F8	This rule indicates the target OS is a pre-release, Windows Insider build, and the target machine has Secure Boot enabled. This rule blocks the pre-release signed build from booting if installed on the machine.
DiskSpaceBlockInDownLevel	6080AFAC-892E-4903-94EA-7A17E69E549E	This failure indicates the system ran out of disk space during the downlevel operations of upgrade.
DiskSpaceFailure	981DCBA5-B8D0-4BA7-A8AB-4030F7A10191	This failure indicates the system drive ran out of available disk space at some point after the first reboot into the upgrade.
PreReleaseWimMountDriverFound	31EC76CC-27EC-4ADC-9869-66AABEDB56F0	Captures failures due to having an unrecognized `wimmount.sys` driver registered on the system.
DebugSetupMemoryDump	C7C63D8A-C5F6-4255-8031-74597773C3C6	This offline only rule indicates a bug check occurred during setup. If the debugger tools are available on the system, SetupDiag debugs the memory dump and provide details.
DebugSetupCrash	CEEBA202-6F04-4BC3-84B8-7B99AED924B1	This offline only rule indicates that setup itself encountered a failure that resulted in a process memory dump. If the debugger tools are installed on the system, SetupDiag debugs the memory dump and give further details.
DebugMemoryDump	505ED489-329A-43F5-	This offline only rule is for any memory.dmp file that resulted during the setup/upgrade

Rule Name	GUID	Description
	B467-FCAAF6A1264C	operation. If the debugger tools are installed on the system, SetupDiag debugs the memory dump and give further details.
DeviceInstallHang	37BB1C3A-4D79-40E8-A556-FDA126D40BC6	This failure rule indicates the system hung or bug checked during the device installation phase of upgrade.
DriverPackageMissingFileFailure	37BB1C3A-4D79-40E8-A556-FDA126D40BC6	This rule indicates that a driver package had a missing file during device install. Updating the driver package might help resolve the issue.
UnsignedDriverBootFailure	CD270AA4-C044-4A22-886A-F34EF2E79469	This rule indicates that an unsigned driver caused a boot failure.
BootFailureDetected	4FB446C2-D4EC-40B4-97E2-67EB19D1CFB7	This rule indicates a boot failure occurred during a specific phase of the update. The rule indicates the failure code and phase for diagnostic purposes.
WinSetupBootFilterFailure	C073BFC8-5810-4E19-B53B-4280B79E096C	Detects failures in the kernel mode file operations.
FindDebugInfoFromRollbackLog	9600EB68-1120-4A87-9FE9-3A4A70ACFC37	This rule determines and gives details when a bug check occurs during the setup/upgrade process that resulted in a memory dump. However, a debugger package isn't required on the executing machine.
AdvancedInstallerFailed	77D36C96-32BE-42A2-BB9C-AAFFE64FCADC	Finds fatal advanced installer operations that cause setup failures. Indicates critical failure in the AdvancedInstaller while running an installer package, includes the .exe being called, the phase, mode, component, and error codes.

Rule Name	GUID	Description
AdvancedInstallerPluginInstallFailed	2F784A0E-CEB1-47C5-8072-F1294C7CB4AE	This rule indicates some component that was being installed via an advanced installer (FeatureOnDemand, Language Packs, .NET packages, etc.) failed to install. The rule calls out what was being installed. If the failed component is a FeatureOnDemand, remove the Windows Feature, reboot, and try the upgrade again. If the failed component is a Language Pack, remove the additional language pack, reboot, and try the upgrade again.
AdvancedInstallerGenericFailure	4019550D-4CAA-45B0-A222-349C48E86F71	A rule to match AdvancedInstaller read/write failures in a generic sense. Triggers on advanced installer failures in a generic sense. It outputs the application called, phase, mode, component, and error code.
FindMigApplyUnitFailure	A4232E11-4043-4A37-9BF4-5901C46FD781	Detects a migration unit failure that caused the update to fail. This rule outputs the name of the migration plug-in and the error code it produced for diagnostic purposes.
FindMigGatherUnitFailure	D04C064B-CD77-4E64-96D6-D26F30B4EE29	Detects a migration gather unit failure that caused the update to fail. This rule outputs the name of the gather unit/plug-in and the error code it produced for diagnostic purposes.
FindMigGatherApplyFailure	A9964E6C-A2A8-45FF-B6B5-25E0BD71428E	Shows errors when the migration Engine fails out on a gather or apply operation. Indicates the Migration Object (file or registry path), the Migration
OptionalComponentFailedToGetOCsFromPackage	D012E2A2-99D8-4A8C-	This rule matches a specific Optional Component failure when attempting to enumerate

Rule Name	GUID	Description
	BBB2-088B92083D78	components in a package. Indicates the optional component (OC) migration operation failed to enumerate optional components from an OC Package. It outputs the package name and error code. This rule replaces the OptionalComponentInstallFailure rule present.
OptionalComponentOpenPackageFailed	22952520-EC89-4FBD-94E0-B67DF88347F6	Matches a specific Optional Component failure when attempting to open an OC package. It outputs the package name and error code. Indicates the optional component migration operation failed to open an optional component Package. Outputs the package name and error code.
OptionalComponentInitCBSSessionFailed	63340812-9252-45F3-A0F2-B2A4CA5E9317	Matches a specific failure where the advanced installer service or components aren't operating or started on the system. Indicates corruption in the servicing stack on the downlevel system. Outputs the error code encountered while trying to initialize the servicing component on the existing OS.
CriticalSafeOSDUFailure	73566DF2-CA26-4073-B34C-C9BC70DBF043	This rule indicates a failure occurred while updating the SafeOS image with a critical dynamic update. It indicates the phase and error code that occurred while attempting to update the SafeOS image for diagnostic purposes.
UserProfileCreationFailureDuringOnlineApply	678117CE-F6A9-40C5-BC9F-A22575C78B14	Indicates there was a critical failure while creating or modifying a User Profile during the online apply phase of the update. It indicates the operation and error code associated with

Rule Name	GUID	Description
		the failure for diagnostic purposes.
UserProfileCreationFailureDuringFinalize	C6677BA6-2E53-4A88-B528-336D15ED1A64	Matches a specific User Profile creation error during the finalize phase of setup. It outputs the failure code.
UserProfileSuffixMismatch	B4BBCCCE-F99D-43EB-9090-078213397FD8	Detects when a file or other object causes the migration or creation of a user profile to fail during the update.
DuplicateUserProfileFailure	BD7B3109-80F1-4421-8F0A-B34CD25F4B51	This rule indicates a fatal error while migrating user profiles, usually with multiple SIDs associated with a single user profile. This error usually occurs when software creates local user accounts that aren't ever used or signed in with. The rule indicates the SID and UserName of the account that is causing the failure. To attempt to resolve the issue, first back up all the user's files for the affected user account. After the user's files are backed up, delete the account in a supported manner. Make sure that the account isn't one that is needed or is currently used to sign into the device. After deleting the account, reboot, and try the upgrade again.
WimMountFailure	BE6DF2F1-19A6-48C6-AEF8-D3B0CE3D4549	This rule indicates the update failed to mount a WIM file. It shows the name of the WIM file and the error message and error code associated with the failure for diagnostic purposes.
WimMountDriverIssue	565B60DD-5403-4797-AE3E-BC5CB972FBAE	Detects failures in `WimMount.sys` registration on the system.
WimApplyExtractFailure	746879E9-C9C5-488C-	Matches a WIM apply failure during WIM extraction phases of

Rule Name	GUID	Description
	8D4B-0C811FF3A9A8	setup. It outputs the extension, path, and error code.
UpdateAgentExpanderFailure	66E496B3-7D19-47FA-B19B-4040B9FD17E2	Matches DPX expander failures in the downlevel phase of update from Windows Update. It outputs the package name, function, expression, and error code.
FindFatalPluginFailure	E48E3F1C-26F6-4AFB-859B-BF637DA49636	Matches any plug-in failure that setupplatform decides is fatal to setup. It outputs the plugin name, operation, and error code.
MigrationAbortedDueToPluginFailure	D07A24F6-5B25-474E-B516-A730085940C9	Indicates a critical failure in a migration plugin that causes setup to abort the migration. Provides the setup operation, plug-in name, plug-in action and error code.
DISMAddPackageFailed	6196FF5B-E69E-4117-9EC6-9C1EAB20A3B9	Indicates a critical failure during a DISM add package operation. Specifies the Package Name, DISM error and add package error code.
DISMImageSessionFailure	61B7886B-10CD-4C98-A299-B987CB24A11C	Captures failure information when DISM fails to start an image session successfully.
DISMproviderFailure	D76EF86F-B3F8-433F-9EBF-B4411F8141F4	Triggers when a DISM provider (plug-in) fails in a critical operation. Outputs the file (plug-in name), function called + error code, and error message from the provider.
SysPrepLaunchModuleFailure	7905655C-F295-45F7-8873-81D6F9149BFD	Indicates a sysPrep plug-in failed in a critical operation. Indicates the plug-in name, operation name, and error code.
UserProvidedDriverInjectionFailure	2247C48A-7EE3-4037-AFAB-95B92DE1D980	A driver provided to setup (via command line input) failed in some way. Outputs the driver install function and error code.

Rule Name	GUID	Description
DriverMigrationFailure	9378D9E2-256E-448C-B02F-137F611F5CE3	This rule indicates a fatal failure when migrating drivers.
UnknownDriverMigrationFailure	D7541B80-5071-42CE-AD14-FBE8C0C4F7FD	This rule indicates a bad driver package resides on the system. The driver package causes the upgrade to fail when the driver package is attempted to migrate to the new OS. The rule usually indicates the driver package name that caused the issue. The remediation is to remove the bad driver package, reboot, and try the upgrade again. If an update to this driver is available from the OEM, updating the driver package is recommended.
FindSuccessfulUpgrade	8A0824C8-A56D-4C55-95A0-22751AB62F3E	Determines if the given setup was a success or not based off the logs.
FindSetupHostReportedFailure	6253C04F-2E4E-4F7A-B88E-95A69702F7EC	Gives information about failures surfaced early in the upgrade process by `setuphost.exe`
FindDownlevelFailure	716334B7-F46A-4BAA-94F2-3E31BC9EFA55	Gives failure information surfaced by SetupPlatform, later in the downlevel phase.
FindAbruptDownlevelFailure	55882B1A-DA3E-408A-9076-23B22A0472BD	Gives last operation failure information when the system fails in the downlevel, but the log just ends abruptly.
FindEarlyDownlevelError	A4CE4FC9-5E10-4BB1-8ECE-3B29EB9D7C52	Detects failures in downlevel phase before setup platform is invoked.

Rule Name	GUID	Description
FindSPFatalError	A4028172-1B09-48F8-AD3B-86CDD7D55852	Captures failure information when setup platform encounters a fatal error.
FindSetupPlatformFailedOperationInfo	307A0133-F06B-4B75-AEA8-116C3B53C2D1	Gives last phase and error information when SetupPlatform indicates a critical failure. This rule indicates the operation and error associated with the failure for diagnostic purposes.
FindRollbackFailure	3A43C9B5-05B3-4F7C-A955-88F991BB5A48	Gives last operation, failure phase, and error information when a rollback occurs.

Sample logs

Text log sample

txt

```
Matching Profile found: OptionalComponentOpenPackageFailed - 22952520-EC89-4FBD-94E0-B67DF88347F6
System Information:
    Machine Name = Offline
    Manufacturer = MSI
    Model = MS-7998
    HostOSArchitecture = x64
    FirmwareType = PCAT
    BiosReleaseDate = 20160727000000.000000+000
    BiosVendor = BIOS Date: 07/27/16 10:01:46 Ver: V1.70
    BiosVersion = 1.70
    HostOSVersion = 10.0.15063
    HostOSBuildString = 15063.0.amd64fre.rs2_release.170317-1834
    TargetOSBuildString = 10.0.16299.15 (rs3_release.170928-1534)
    HostOSLanguageId = 2057
    HostOSEdition = Core
    RegisteredAV = Windows Defender,
    FilterDrivers = WdFilter,wcifs,WIMMount,luafv,Wof,FileInfo,
    UpgradeStartTime = 3/21/2023 9:47:16 PM
    UpgradeEndTime = 3/21/2023 10:02:40 PM
    UpgradeElapsedTime = 00:15:24
    ReportId = dd4db176-4e3f-4451-aef6-22cf46de8bde

Error: SetupDiag reports Optional Component installation failed to open OC Package.
```

```
Package Name: Foundation, Error: 0x8007001F
Recommend you check the "Windows Modules Installer" service (Trusted Installer) is
started on the system and set to automatic start, reboot and try the update again.
Optionally, you can check the status of optional components on the system (search
for Windows Features), uninstall any unneeded optional components, reboot and try
the update again.
Error: SetupDiag reports downlevel failure, Operation: Finalize, Error: 0x8007001F -
0x50015
Refer to https://learn.microsoft.com/windows/deployment/upgrade/upgrade-error-codes
for error information.
```

XML log sample

XML


```xml
<?xml version="1.0" encoding="utf-16"?>
<SetupDiag xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="https://learn.microsoft.com/windows/deployment/upgrade/setupdiag">
  <Version>1.7.0.0</Version>
  <ProfileName>FindSPFatalError</ProfileName>
  <ProfileGuid>A4028172-1B09-48F8-AD3B-86CDD7D55852</ProfileGuid>
  <SystemInfo>
    <MachineName>Offline</MachineName>
    <Manufacturer>Gigabyte Technology Co., Ltd.</Manufacturer>
    <Model>X470 AORUS ULTRA GAMING</Model>
    <HostOSArchitecture>1033</HostOSArchitecture>
    <FirmwareType>UEFI</FirmwareType>
    <BiosReleaseDate>20180808000000.000000+000</BiosReleaseDate>
    <BiosVendor>F3</BiosVendor>
    <BiosVersion />
    <HostOSVersion>10.0.18908</HostOSVersion>
    <HostOSBuildString>18908.1000.amd64fre.rs_prerelease.190524-
1658</HostOSBuildString>
    <TargetOSBuildString>10.0.18912.1001 (rs_prerelease.190601-1739)
</TargetOSBuildString>
    <HostOSLanguageId />
    <HostOSEdition>Professional</HostOSEdition>
    <RegisteredAV>Windows Defender</RegisteredAV>
    <FilterDrivers />
    <UpgradeStartTime>2023-06-06T21:19:10</UpgradeStartTime>
    <UpgradeElapsedTime />
    <UpgradeEndTime>2023-06-06T22:21:49</UpgradeEndTime>
    <RollbackStartTime>0001-01-01T00:00:00</RollbackStartTime>
    <RollbackEndTime>0001-01-01T00:00:00</RollbackEndTime>
    <RollbackElapsedTime />
    <FinalizeStartTime>0001-01-01T00:00:00</FinalizeStartTime>
    <PostOOBESuccessTime>0001-01-01T00:00:00</PostOOBESuccessTime>
    <TotalOfflineTime />
    <CommercialId>Offline</CommercialId>
    <CV>MgUweCZk90KdwUiZ</CV>
    <SetupReportId>F21F8FB6-00FD-4349-84FB-2AC75F389E73</SetupReportId>
    <ReportId>F21F8FB6-00FD-4349-84FB-2AC75F389E73</ReportId>
```

```
  </SystemInfo>
  <LogErrorLine>2023-06-06 21:47:11, Error                 SP     Error converting
install time 5/2/2023 to structure[gle=0x00000057]</LogErrorLine>
  <FailureData>
Error: SetupDiag reports Fatal Error.
Last Setup Phase = Downlevel
Last Setup Operation: Gather data, scope: EVERYTHING
Error: 0x00000057</FailureData>
  <FailureData>LogEntry: 2023-06-06 21:47:11, Error                 SP     Error
converting install time 5/2/2023 to structure[gle=0x00000057]</FailureData>
  <FailureData>LogEntry: 2023-06-06 21:47:11, Error                 SP     Error
converting install time 5/2/2023 to structure[gle=0x00000057]</FailureData>
  <FailureData>
Refer to "https://learn.microsoft.com/windows/desktop/Debug/system-error-codes" for
error information.</FailureData>
  <FailureDetails>Err = 0x00000057, LastOperation = Gather data, scope: EVERYTHING,
LastPhase = Downlevel</FailureDetails>
</SetupDiag>
```

JSON log sample

JSON


```
{
    "Version":"1.6.0.0",
    "ProfileName":"FindSPFatalError",
    "ProfileGuid":"A4028172-1B09-48F8-AD3B-86CDD7D55852",
    "SystemInfo":{
        "BiosReleaseDate":"20180808000000.000000+000",
        "BiosVendor":"F3",
        "BiosVersion":"F3",
        "CV":"MgUweCZk90KdwUiZ",
        "CommercialId":"Offline",
        "FilterDrivers":"",
        "FinalizeStartTime":"\/Date(-62135568000000-0800)\/",
        "FirmwareType":"UEFI",
        "HostOSArchitecture":"x64",
        "HostOSBuildString":"18908.1000.amd64fre.rs_prerelease.190524-1658",
        "HostOSEdition":"Professional",
        "HostOSLanguageId":"",
        "HostOSVersion":"",
        "MachineName":"Offline",
        "Manufacturer":"Gigabyte Technology Co., Ltd.",
        "Model":"X470 AORUS ULTRA GAMING",
        "PostOOBESuccessTime":"\/Date(-62135568000000-0800)\/",
        "RegisteredAV":"Windows Defender",
        "ReportId":"F21F8FB6-00FD-4349-84FB-2AC75F389E73",
        "RollbackElapsedTime":"PT0S",
        "RollbackEndTime":"\/Date(-62135568000000-0800)\/",
        "RollbackStartTime":"\/Date(-62135568000000-0800)\/",
        "SetupReportId":"F21F8FB6-00FD-4349-84FB-2AC75F389E73",
        "TargetOSArchitecture":null,
        "TargetOSBuildString":"10.0.18912.1001 (rs_prerelease.190601-1739)",
```

```
        "TotalOfflineTime":"PT0S",
        "UpgradeElapsedTime":"PT1H2M39S",
        "UpgradeEndTime":"\/Date(1559884909000-0700)\/",
        "UpgradeStartTime":"\/Date(1559881150000-0700)\/"
    },
    "LogErrorLine":"2023-06-06 21:47:11, Error                 SP     Error
converting install time 5\/2\/2023 to structure[
        gle=0x00000057
    ]",
    "FailureData":[
        "\u000aError: SetupDiag reports Fatal Error.\u000aLast Setup Phase =
Downlevel\u000aLast Setup Operation: Gather data, scope: EVERYTHING\u000aError:
0x00000057",
        "LogEntry: 2023-06-06 21:47:11, Error                 SP     Error
converting install time 5\/2\/2023 to structure[
            gle=0x00000057
        ]",
        "LogEntry: 2023-06-06 21:47:11, Error                 SP     Error
converting install time 5\/2\/2023 to structure[
            gle=0x00000057
        ]",
        "\u000aRefer to
\"https:\/\/learn.microsoft.com\/windows\/desktop\/Debug\/system-error-codes\" for
error information."
    ],
    "FailureDetails":"Err = 0x00000057, LastOperation = Gather data, scope:
EVERYTHING, LastPhase = Downlevel",
    "DeviceDriverInfo":null,
    "Remediation":[

    ],
    "SetupPhaseInfo":null,
    "SetupOperationInfo":null
}
```


Example registry key

Related articles

- Resolve Windows upgrade errors: Technical information for IT Pros.

Last updated on 04/08/2025

Windows 10 upgrade issues troubleshooting

01/15/2025Applies to: Supported versions of Windows Client

Try our Virtual Agent - It can help you quickly identify and fix common Windows boot issues

> **Note**
>
> This is a 300 level topic (moderately advanced).
>
> For IT professionals, check more information in Resolve Windows 10 upgrade errors.

If a Windows 10 upgrade isn't successful, it can be helpful to understand when an error occurred in the upgrade process.

> **Important**
>
> Use the SetupDiag tool before you begin manually troubleshooting an upgrade error. SetupDiag automates log file analysis, detecting and reporting details on many different types of known upgrade issues.

Applies to: Windows 10

Actions performed during upgrade processes

Briefly, the upgrade process consists of four phases that are controlled by Windows Setup: Downlevel, SafeOS, First boot, and Second boot. The computer will reboot once between each phase. Note: Progress is tracked in the registry during the upgrade process using the following key: `HKLM\System\Setup\mosetup\volatile\SetupProgress`. This key is volatile and only present during the upgrade process; it contains a binary value in the range 0-100.

These phases are explained in greater detail below. First, let's summarize the actions performed during each phase because this affects the type of errors that can be encountered.

1. Downlevel phase: Because this phase runs on the source OS, upgrade errors aren't typically seen. If you do encounter an error, ensure the source OS is stable. Also ensure the Windows setup source and the destination drive are accessible.

2. SafeOS phase: Errors most commonly occur during this phase due to hardware issues, firmware issues, or non-microsoft disk encryption software.

   Since the computer is booted into Windows PE during the SafeOS phase, a useful troubleshooting technique is to boot into Windows PE using installation media. You can use the media creation tool to create bootable media, or you can use tools such as the Windows ADK, and then boot your device from this media to test for hardware and firmware compatibility issues.

   > **Tip**
   >
   > If you attempt to use the media creation tool with a USB drive and this fails with error 0x80004005 - 0xa001a, this is because the USB drive is using GPT partition style. The tool requires that you use MBR partition style. You can use the `DISKPART` command to convert the USB drive from GPT to MBR. For more information, see Change a GUID Partition Table Disk into a Master Boot Record Disk.

   Don't proceed with the Windows 10 installation after booting from this media. This method can only be used to perform a clean install, which won't migrate any of your apps and settings, and you'll be required reenter your Windows 10 license information.

   If the computer doesn't successfully boot into Windows PE using the media that you created, this is likely due to a hardware or firmware issue. Check with your hardware manufacturer and apply any recommended BIOS and firmware updates. If you're still unable to boot to installation media after applying updates, disconnect or replace legacy hardware.

   If the computer successfully boots into Windows PE, but you are not able to browse the system drive on the computer, it's possible that non-Microsoft disk encryption software is blocking your ability to perform a Windows 10 upgrade. Update or temporarily remove the disk encryption.

3. First boot phase: Boot failures in this phase are relatively rare, and almost exclusively caused by device drivers. Disconnect all peripheral devices except for the mouse, keyboard, and display. Obtain and install updated device drivers, then retry the upgrade.

4. Second boot phase: In this phase, the system is running under the target OS with new drivers. Boot failures are most commonly due to anti-virus software or filter drivers. Disconnect all peripheral devices except for the mouse, keyboard, and display. Obtain and install updated device drivers, temporarily uninstall anti-virus software, then retry the upgrade.

If the general troubleshooting techniques described above or the quick fixes detailed below don't resolve your issue, you can attempt to analyze log files and interpret upgrade error codes. You can also Submit Windows 10 upgrade errors using Feedback Hub so that Microsoft can diagnose your issue.

The Windows 10 upgrade process

The Windows Setup application is used to upgrade a computer to Windows 10, or to perform a clean installation. Windows Setup starts and restarts the computer, gathers information, copies files, and creates or adjusts configuration settings.

When performing an operating system upgrade, Windows Setup uses phases described below. A reboot occurs between each of the phases. After the first reboot, the user interface will remain the same until the upgrade is completed. Percent progress is displayed and will advance as you move through each phase, reaching 100% at the end of the second boot phase.

1. Downlevel phase: The downlevel phase is run within the previous operating system. Windows files are copied and installation components are gathered.

2. Safe OS phase: A recovery partition is configured, Windows files are expanded, and updates are installed. An OS rollback is prepared if needed. Example error codes: 0x2000C, 0x20017.

3. First boot phase: Initial settings are applied. Example error codes: 0x30018, 0x3000D.

4. Second boot phase: Final settings are applied. This is also called the OOBE boot phase. Example error codes: 0x4000D, 0x40017.

At the end of the second boot phase, the **Welcome to Windows 10** screen is displayed, preferences are configured, and the Windows 10 sign-in prompt is displayed.

5. Uninstall phase: This phase occurs if upgrade is unsuccessful (image not shown). Example error codes: 0x50000, 0x50015.

   Figure 1: Phases of a successful Windows 10 upgrade (uninstall isn't shown):

   DU = Driver/device updates. OOBE = Out of box experience. WIM = Windows image (Microsoft)

Data collection

If you need assistance from Microsoft support, we recommend you collect the information by following the steps mentioned in Gather information by using TSS for deployment-related issues.

More information

- Windows 10 FAQ for IT professionals
- Windows 10 Enterprise system requirements
- Windows 10 Specifications
- Windows 10 IT pro forums
- Fix Windows Update errors by using the DISM or System Update Readiness tool

Windows Error Reporting

Applies to: ✅ Windows 11, ✅ Windows 10

> **Note**
>
> This article is a 300 level article (moderately advanced).
>
> See Resolve Windows upgrade errors for a full list of articles in this section.

When Windows Setup fails, the result code and extend code are recorded as an informational event in the Application log by Windows Error Reporting as event 1001. The event name is **WinSetupDiag02**. Event Viewer or Windows PowerShell can be used to review this event.

To use Windows PowerShell, type the following commands from an elevated Windows PowerShell prompt:

> **Important**
>
> The following Event logs are only available if Windows was updated from a previous version of Windows to a new version of Windows.

PowerShell

```
$events = Get-WinEvent -FilterHashtable
@{LogName="Application";ID="1001";Data="WinSetupDiag02"}
$event = [xml]$events[0].ToXml()
$event.Event.EventData.Data
```

To use Event Viewer:

1. Open Event Viewer and navigate to **Windows Logs\Application**.
2. Select **Find**, and then search for **winsetupdiag02**.
3. Double-click the event that is highlighted.

> **Note**
>
> For legacy operating systems, the Event Name was WinSetupDiag01.

Ten parameters are listed in the event:

Expand table

Parameters	Description	Example
P1	The Setup Scenario	1=Media, 5=WindowsUpdate, 7=Media Creation Tool
P2	Setup Mode	x=default, 1=Downlevel, 5=Rollback
P3	New OS Architecture	x=default, 0=X86, 9=AMD64
P4	Install Result	x=default, 0=Success, 1=Failure,2=Cancel, 3=Blocked
P5	Result Error Code	0xc1900101
P6	Extend Error Code	0x20017
P7	Source OS build	9600
P8	Source OS branch	Not typically available
P9	New OS build	16299
P10	New OS branch	rs3_release

The event also contains links to log files that can be used to perform a detailed diagnosis of the error. The following example is an example of this event from a successful upgrade:

Event Properties - Event 1001, Windows Error Reporting

General | Details

```
Fault bucket , type 0
Event Name: WinSetupDiag02
Response: Not available
Cab Id: 0

Problem signature:
P1: 1
P2: 4
P3: 9
P4: 0
P5: 0x0
P6: 0x0
P7: 7601
P8: win7sp1_rtm
P9: 16299
P10: rs3_release

Attached files:
\\?\C:\Windows\Panther\SetupAct.log
\\?\C:\Windows\Panther\diagerr.xml
\\?\C:\Windows\inf\setupapi.setup.log
\\?\C:\Windows\inf\setupapi.dev.log
\\?\C:\Windows\inf\setupapi.offline.log
\\?\C:\ProgramData\Microsoft\Windows\WER\Temp\WER8A0F.tmp.WERInternalMetadata.xml
\\?\C:\ProgramData\Microsoft\Windows\WER\Temp\WER8BD4.tmp.csv
\\?\C:\ProgramData\Microsoft\Windows\WER\Temp\WER8C04.tmp.txt

These files may be available here:
C:\ProgramData\Microsoft\Windows\WER\ReportQueue\NonCritical_1
_d2dd2fd6e01459e92a498934e219ae428bafdcaa_00000000_cab_08928c51

Analysis symbol:
Rechecking for solution: 0
Report Id: d53897e2-49f6-4399-b9c2-17adb556149f
Report Status: 4
Hashed bucket:
```

Log Name:	Application		
Source:	Windows Error Reporting	Logged:	1/31/2018 11:39:55 AM
Event ID:	1001	Task Category:	None
Level:	Information	Keywords:	Classic
User:	N/A	Computer:	user1-PC
OpCode:			
More Information:	Event Log Online Help		

Copy | Close

Related articles

- Fix Windows Update errors by using the DISM or System Update Readiness tool.

Last updated on 04/08/2025

Windows 10 upgrade error codes

01/15/2025Applies to: Supported versions of Windows Client

Applies to: Windows 10

> **Note**
>
> This is a 400 level topic (advanced).
>
> For IT professionals, check more information in Resolve Windows 10 upgrade errors.

If the upgrade process isn't successful, Windows Setup will return two codes:

1. A result code: The result code corresponds to a specific Win32 or NTSTATUS error.
2. An extend code: The extend code contains information about both the phase in which an error occurred, and the operation that was being performed when the error occurred.

For example, a result code of 0xC1900101 with an extend code of 0x4000D will be returned as: 0xC1900101 - 0x4000D.

> **Note**
>
> If only a result code is returned, this can be because a tool is being used that was not able to capture the extend code. For example, if you are using the Windows 10 Upgrade Assistant then only a result code might be returned.

> **Tip**
>
> If you are unable to locate the result and extend error codes, you can attempt to find these codes using Event Viewer. For more information, see Windows Error Reporting.

Result codes

A result code of 0xC1900101 is generic and indicates that a rollback occurred. In most cases, the cause is a driver compatibility issue. To troubleshoot a failed upgrade that has returned a result code of 0xC1900101, analyze the extend code to determine the Windows Setup phase, and see the Resolution procedures section later in this article.

The following set of result codes is associated with Windows Setup compatibility warnings:

Expand table

Result code	Message	Description
0xC1900210	MOSETUP_E_COMPAT_SCANONLY	Setup didn't find any compat issue
0xC1900208	MOSETUP_E_COMPAT_INSTALLREQ_BLOCK	Setup found an actionable compat issue, such as an incompatible app
0xC1900204	MOSETUP_E_COMPAT_MIGCHOICE_BLOCK	The migration choice selected isn't available (ex: Enterprise to Home)
0xC1900200	MOSETUP_E_COMPAT_SYSREQ_BLOCK	The computer isn't eligible for Windows 10
0xC190020E	MOSETUP_E_INSTALLDISKSPACE_BLOCK	The computer doesn't have enough free space to install

A list of modern setup (mosetup) errors with descriptions in the range is available in the Resolution procedures section in this article.

Other result codes can be matched to the specific type of error encountered. To match a result code to an error:

1. Identify the error code type as either Win32 or NTSTATUS using the first hexadecimal digit:

   8 = Win32 error code (ex: 0x80070070) C = NTSTATUS value (ex: 0xC1900107)

2. Write down the last four digits of the error code (ex: 0x80070070 = 0070). These digits are the actual error code type as defined in the HRESULT or the NTSTATUS structure. Other digits in the code identify things such as the device type that produced the error.

3. Based on the type of error code determined in the first step (Win32 or NTSTATUS), match the four digits derived from the second step to either a Win32 error code or NTSTATUS value using the following links:

   - Win32 error code
   - NTSTATUS value

Examples:

- 0x80070070

  1. Based on the "8", this is a Win32 error code.
  2. The last four digits are 0070, so look up 0x00000070 in the Win32 error code table.
  3. The error is:

> ERROR_DISK_FULL

- 0xC1900107

  1. Based on the "C", this is an NTSTATUS error code.
  2. The last four digits are 0107, so look up 0x00000107 in the NTSTATUS value table.
  3. The error is:

  > STATUS_SOME_NOT_MAPPED

Some result codes are self-explanatory, whereas others are more generic and require further analysis. In the examples shown above, ERROR_DISK_FULL indicates that the hard drive is full and extra room is needed to complete Windows upgrade. The message STATUS_SOME_NOT_MAPPED is more ambiguous, and means that an action is pending. In this case, the action pending is often the cleanup operation from a previous installation attempt, which can be resolved with a system reboot.

Extend codes

> **Important**
>
> Extend codes reflect the current Windows 10 upgrade process, and might change in future releases of Windows 10. The codes discussed in this section apply to Windows 10 version 1607, also known as the Anniversary Update.

Extend codes can be matched to the phase and operation when an error occurred. To match an extend code to the phase and operation:

1. Use the first digit to identify the phase (ex: 0x4000D = 4).
2. Use the last two digits to identify the operation (ex: 0x4000D = 0D).
3. Match the phase and operation to values in the tables provided below.

The following tables provide the corresponding phase and operation for values of an extend code:

Extend code: phase

Expand table

Hex	Phase
0	SP_EXECUTION_UNKNOWN

Hex	Phase
1	SP_EXECUTION_DOWNLEVEL
2	SP_EXECUTION_SAFE_OS
3	SP_EXECUTION_FIRST_BOOT
4	SP_EXECUTION_OOBE_BOOT
5	SP_EXECUTION_UNINSTALL

Extend code: operation

Expand table

Hex	Operation
0	SP_EXECUTION_OP_UNKNOWN
1	SP_EXECUTION_OP_COPY_PAYLOAD
2	SP_EXECUTION_OP_DOWNLOAD_UPDATES
3	SP_EXECUTION_OP_INSTALL_UPDATES
4	SP_EXECUTION_OP_INSTALL_RECOVERY_ENVIRONMENT
5	SP_EXECUTION_OP_INSTALL_RECOVERY_IMAGE
6	SP_EXECUTION_OP_REPLICATE_OC
7	SP_EXECUTION_OP_INSTALL_DRIVERS
8	SP_EXECUTION_OP_PREPARE_SAFE_OS
9	SP_EXECUTION_OP_PREPARE_ROLLBACK
A	SP_EXECUTION_OP_PREPARE_FIRST_BOOT
B	SP_EXECUTION_OP_PREPARE_OOBE_BOOT
C	SP_EXECUTION_OP_APPLY_IMAGE
D	SP_EXECUTION_OP_MIGRATE_DATA
E	SP_EXECUTION_OP_SET_PRODUCT_KEY
F	SP_EXECUTION_OP_ADD_UNATTEND

Expand table

Hex	Operation
10	SP_EXECUTION_OP_ADD_DRIVER
11	SP_EXECUTION_OP_ENABLE_FEATURE
12	SP_EXECUTION_OP_DISABLE_FEATURE
13	SP_EXECUTION_OP_REGISTER_ASYNC_PROCESS
14	SP_EXECUTION_OP_REGISTER_SYNC_PROCESS
15	SP_EXECUTION_OP_CREATE_FILE
16	SP_EXECUTION_OP_CREATE_REGISTRY
17	SP_EXECUTION_OP_BOOT
18	SP_EXECUTION_OP_SYSPREP
19	SP_EXECUTION_OP_OOBE
1A	SP_EXECUTION_OP_BEGIN_FIRST_BOOT
1B	SP_EXECUTION_OP_END_FIRST_BOOT
1C	SP_EXECUTION_OP_BEGIN_OOBE_BOOT
1D	SP_EXECUTION_OP_END_OOBE_BOOT
1E	SP_EXECUTION_OP_PRE_OOBE
1F	SP_EXECUTION_OP_POST_OOBE
20	SP_EXECUTION_OP_ADD_PROVISIONING_PACKAGE

For example: An extend code of 0x4000D, represents a problem during phase 4 (0x4) with data migration (000D).

Data collection

If you need assistance from Microsoft support, we recommend you collect the information by following the steps mentioned in Gather information by using TSS for deployment-related issues.

More information

- Windows 10 FAQ for IT professionals
- Windows 10 Enterprise system requirements
- Windows 10 Specifications
- Microsoft Windows Q & A
- Fix Windows Update errors by using the DISM or System Update Readiness tool

Windows upgrade log files

Applies to: ✅ Windows 11, ✅ Windows 10

> **Note**
>
> This article is a 400-level article (advanced).
>
> See Resolve Windows upgrade errors for a full list of articles in this section.

Several log files are created during each phase of the upgrade process. These log files are essential for troubleshooting upgrade problems. By default, the folders that contain these log files are hidden on the upgrade target computer. To view the log files, configure Windows Explorer to view hidden items, or use a tool to automatically gather these logs. The most useful log is **setupact.log**. The log files are located in a different folder depending on the Windows Setup phase. Recall that the phase can be determined from the extend code.

> **Note**
>
> Also see the Windows Error Reporting article in this section for help with locating error codes and log files.

The following table describes some log files and how to use them for troubleshooting purposes:

Expand table

Log file	Phase: Location	Description	When to use
setupact.log	Down-Level: $Windows.~BT\Sources\Panther	Contains information about setup actions during the downlevel phase.	All downlevel failures and starting point for rollback investigations. Setup.act is the most important log for diagnosing setup issues.
setupact.log	Out of box experience (OOBE): $Windows.~BT\Sources\Panther\UnattendGC	Contains information about actions during the OOBE phase.	Investigating rollbacks that failed during OOBE phase and operations -

Log file	Phase: Location	Description	When to use
			0x4001C, 0x4001D, 0x4001E, 0x4001F.
setupact.log	Rollback: $Windows.~BT\Sources\Rollback	Contains information about actions during rollback.	Investigating generic rollbacks - 0xC1900101.
setupact.log	Pre-initialization (before downlevel): Windows	Contains information about initializing setup.	If setup fails to launch.
setupact.log	Post-upgrade (after OOBE): Windows\Panther	Contains information about setup actions during the installation.	Investigate post-upgrade related issues.
setuperr.log	Same as setupact.log	Contains information about setup errors during the installation.	Review all errors encountered during the installation phase.
miglog.xml	Post-upgrade (after OOBE): Windows\Panther	Contains information about what was migrated during the installation.	Identify post upgrade data migration issues.
BlueBox.log	Down-Level: Windows\Logs\Mosetup	Contains information communication between `setup.exe` and Windows Update.	Use during WSUS and Windows Update downlevel failures or for 0xC1900107.
Supplemental rollback logs: **Setupmem.dmp** **setupapi.dev.log** Event logs (*.evtx)	$Windows.~BT\Sources\Rollback	Additional logs collected during rollback.	Setupmem.dmp: If OS bug checks during upgrade, setup attempts to extract a mini-dump. Setupapi: Device install issues - 0x30018

Log file	Phase: Location	Description	When to use
			Event logs: Generic rollbacks (0xC1900101) or unexpected reboots.

Log entry structure

A `setupact.log` or `setuperr.log` entry includes the following elements:

1. **The date and time** - 2023-09-08 09:20:05
2. **The log level** - Info, Warning, Error, Fatal Error
3. **The logging component** - CONX, MOUPG, PANTHR, SP, IBSLIB, MIG, DISM, CSI, CBS

The logging components SP (setup platform), MIG (migration engine), and CONX (compatibility information) are useful for troubleshooting Windows Setup errors.

1. **The message** - Operation completed successfully.

See the following example:

Expand table

Date/Time	Log level	Component	Message
2023-09-08 09:23:50,	Warning	MIG	Couldn't replace object C:\Users\name\Cookies. Target Object can't be removed.

Analyze log files

The following instructions are meant for IT professionals. To become more familiar with result codes and extend codes, see the article Upgrade error codes.

To analyze Windows Setup log files:

1. Determine the Windows Setup error code. Windows Setup should return an error code if it isn't successful with the upgrade process.
2. Based on the extend code portion of the error code, determine the type and location of a log file to investigate.

3. Open the log file in a text editor, such as Notepad.

4. Using the result code portion of the Windows Setup error code, search for the result code in the file and find the last occurrence of the code. Alternatively search for the "abort" and abandoning" text strings described in step 7 below.

5. To find the last occurrence of the result code:
   a. Scroll to the bottom of the file and select after the last character.
   b. Select **Edit**.
   c. Select **Find**.
   d. Type the result code.
   e. Under **Direction** select **Up**.
   f. Select **Find Next**.

6. When the last occurrence of the result code is located, scroll up a few lines from this location in the file and review the processes that failed before generating the result code.

7. Search for the following important text strings:

   - `Shell application requested abort`
   - `Abandoning apply due to error for object`

8. Decode Win32 errors that appear in this section.

9. Write down the timestamp for the observed errors in this section.

10. Search other log files for additional information matching these timestamps or errors.

For example, assume that the error code for an error is **0x8007042B - 0x2000D**. Searching for **8007042B** reveals the following content from the `setuperr.log` file:

> ⓘ **Note**
>
> Some lines in the following text are shortened to enhance readability. For example
>
> - The date and time at the start of each line (ex: 2023-10-05 15:27:08) is shortened to minutes and seconds
> - The certificate file name, which is a long text string, is shortened to just "CN."

setuperr.log content:

Console

```
27:08, Error           SP     Error READ, 0x00000570 while gathering/applying
object: File, C:\ProgramData\Microsoft\Crypto\RSA\S-1-5-18 [CN]. Will return
0[gle=0x00000570]
27:08, Error           MIG    Error 1392 while gathering object
C:\ProgramData\Microsoft\Crypto\RSA\S-1-5-18 [CN]. Shell application requested
abort![gle=0x00000570]
27:08, Error                  Gather failed. Last error: 0x00000000
27:08, Error           SP     SPDoFrameworkGather: Gather operation failed. Error:
0x0000002C
27:09, Error           SP     CMigrateFramework: Gather framework failed. Status: 44
27:09, Error           SP     Operation failed: Migrate framework (Full). Error:
0x8007042B[gle=0x000000b7]
27:09, Error           SP     Operation execution failed: 13. hr =
0x8007042B[gle=0x000000b7]
27:09, Error           SP     CSetupPlatformPrivate::Execute: Execution of
operations queue failed, abandoning. Error: 0x8007042B[gle=0x000000b7]
```

The first line indicates there was an error **0x00000570** with the file **C:\ProgramData\Microsoft\Crypto\RSA\S-1-5-18 [CN]**:

Console

```
27:08, Error           SP     Error READ, 0x00000570 while gathering/applying
object: File, C:\ProgramData\Microsoft\Crypto\RSA\S-1-5-18 [CN]. Will return
0[gle=0x00000570]
```

The error **0x00000570** is a Win32 error code corresponding to: **ERROR_FILE_CORRUPT: The file or directory is corrupted and unreadable**.

Therefore, Windows Setup failed because it wasn't able to migrate the corrupt file **C:\ProgramData\Microsoft\Crypto\RSA\S-1-5-18[CN]**. This file is a local system certificate and can be safely deleted. After the `setupact.log` file is searched for more details, the phrase **Shell application requested abort** is found in a location with the same timestamp as the lines in `setuperr.log`. This analysis confirms the suspicion that this file is the cause of the upgrade failure:

setupact.log content:

Console

```
27:00, Info                   Gather started at 10/5/2023 23:27:00
27:00, Info [0x080489] MIG    Setting system object filter context (System)
27:00, Info [0x0803e5] MIG    Not unmapping HKCU\Software\Classes; it is not mapped
27:00, Info [0x0803e5] MIG    Not unmapping HKCU; it is not mapped
27:00, Info            SP     ExecuteProgress: Elapsed events:1 of 4, Percent: 12
27:00, Info [0x0802c6] MIG    Processing GATHER for migration unit:
&lt;System&gt;\UpgradeFramework (CMXEAgent)
27:08, Error           SP     Error READ, 0x00000570 while gathering/applying
object: File, C:\ProgramData\Microsoft\Crypto\RSA\S-1-5-18 [CN]. Will return
```

```
0[gle=0x00000570]
27:08, Error           MIG    Error 1392 while gathering object
C:\ProgramData\Microsoft\Crypto\RSA\S-1-5-18 [CN]. Shell application requested
abort![gle=0x00000570]
27:08, Info            SP     ExecuteProgress: Elapsed events:2 of 4, Percent: 25
27:08, Info            SP     ExecuteProgress: Elapsed events:3 of 4, Percent: 37
27:08, Info [0x080489] MIG    Setting system object filter context (System)
27:08, Info [0x0803e5] MIG    Not unmapping HKCU\Software\Classes; it is not mapped
27:08, Info [0x0803e5] MIG    Not unmapping HKCU; it is not mapped
27:08, Info            MIG    COutOfProcPluginFactory::FreeSurrogateHost: Shutdown
in progress.
27:08, Info            MIG
COutOfProcPluginFactory::LaunchSurrogateHost::CommandLine: -shortened-
27:08, Info            MIG    COutOfProcPluginFactory::LaunchSurrogateHost:
Successfully launched host and got control object.
27:08, Error                  Gather failed. Last error: 0x00000000
27:08, Info                   Gather ended at 10/5/2023 23:27:08 with result 44
27:08, Info                   Leaving MigGather method
27:08, Error           SP     SPDoFrameworkGather: Gather operation failed. Error:
0x0000002C
```

setupapi.dev.log content:

Console

```
>>>  [Device Install (UpdateDriverForPlugAndPlayDevices) - PCI\VEN_8086&DEV_8C4F]
>>>  Section start 2023/09/26 20:13:01.623
      cmd: rundll32.exe
"C:\WINDOWS\Installer\MSI6E4C.tmp",zzzzInvokeManagedCustomActionOutOfProc
SfxCA_95972906 484
ChipsetWiX.CustomAction!Intel.Deployment.ChipsetWiX.CustomActions.InstallDrivers
     ndv: INF path: C:\WINDOWS\TEMP\{15B1CD41-69F5-48EA-9F45-
0560A40FE2D8}\Drivers\lynxpoint\LynxPointSystem.inf
     ndv: Install flags: 0x00000000
     ndv: {Update Device Driver -
PCI\VEN_8086&DEV_8C4F&SUBSYS_05BE1028&REV_04\3&11583659&0&F8}
     ndv:      Search options: 0x00000081
     ndv:      Searching single INF 'C:\WINDOWS\TEMP\{15B1CD41-69F5-48EA-9F45-
0560A40FE2D8}\Drivers\lynxpoint\LynxPointSystem.inf'
     dvi:      {Build Driver List} 20:13:01.643
     dvi:           Searching for hardware ID(s):
     dvi:                pci\ven_8086&dev_8c4f&subsys_05be1028&rev_04
     dvi:                pci\ven_8086&dev_8c4f&subsys_05be1028
     dvi:                pci\ven_8086&dev_8c4f&cc_060100
     dvi:                pci\ven_8086&dev_8c4f&cc_0601
     dvi:           Searching for compatible ID(s):
     dvi:                pci\ven_8086&dev_8c4f&rev_04
     dvi:                pci\ven_8086&dev_8c4f
     dvi:                pci\ven_8086&cc_060100
     dvi:                pci\ven_8086&cc_0601
     dvi:                pci\ven_8086
     dvi:                pci\cc_060100
     dvi:                pci\cc_0601
     sig:           {_VERIFY_FILE_SIGNATURE} 20:13:01.667
```

```
     sig:                Key      = lynxpointsystem.inf
     sig:                FilePath = c:\windows\temp\{15b1cd41-69f5-48ea-9f45-
0560a40fe2d8}\drivers\lynxpoint\lynxpointsystem.inf
     sig:                Catalog  = c:\windows\temp\{15b1cd41-69f5-48ea-9f45-
0560a40fe2d8}\drivers\lynxpoint\LynxPoint.cat
     sig:                Success: File is signed in catalog.
     sig:           {_VERIFY_FILE_SIGNATURE exit(0x00000000)} 20:13:01.683
     dvi:           Created Driver Node:
     dvi:                HardwareID   - PCI\VEN_8086&DEV_8C4F
     dvi:                InfName      - c:\windows\temp\{15b1cd41-69f5-48ea-9f45-
0560a40fe2d8}\drivers\lynxpoint\lynxpointsystem.inf
     dvi:                DevDesc      - Intel(R) QM87 LPC Controller - 8C4F
     dvi:                Section      - Needs_ISAPNP_DRV
     dvi:                Rank         - 0x00ff2001
     dvi:                Signer Score - WHQL
     dvi:                DrvDate      - 04/04/2016
     dvi:                Version      - 10.1.1.18
     dvi:      {Build Driver List - exit(0x00000000)} 20:13:01.699
     ndv:      Searching currently installed INF
     dvi:      {Build Driver List} 20:13:01.699
     dvi:           Searching for hardware ID(s):
     dvi:                pci\ven_8086&dev_8c4f&subsys_05be1028&rev_04
     dvi:                pci\ven_8086&dev_8c4f&subsys_05be1028
     dvi:                pci\ven_8086&dev_8c4f&cc_060100
     dvi:                pci\ven_8086&dev_8c4f&cc_0601
     dvi:           Searching for compatible ID(s):
     dvi:                pci\ven_8086&dev_8c4f&rev_04
     dvi:                pci\ven_8086&dev_8c4f
     dvi:                pci\ven_8086&cc_060100
     dvi:                pci\ven_8086&cc_0601
     dvi:                pci\ven_8086
     dvi:                pci\cc_060100
     dvi:                pci\cc_0601
     dvi:           Created Driver Node:
     dvi:                HardwareID   - PCI\VEN_8086&DEV_8C4F
     dvi:                InfName      -
C:\WINDOWS\System32\DriverStore\FileRepository\lynxpointsystem.inf_amd64_cd1e518d883
ecdfe\lynxpointsystem.inf
     dvi:                DevDesc      - Intel(R) QM87 LPC Controller - 8C4F
     dvi:                Section      - Needs_ISAPNP_DRV
     dvi:                Rank         - 0x00ff2001
     dvi:                Signer Score - WHQL
     dvi:                DrvDate      - 10/03/2016
     dvi:                Version      - 10.1.1.38
     dvi:      {Build Driver List - exit(0x00000000)} 20:13:01.731
     dvi:      {DIF_SELECTBESTCOMPATDRV} 20:13:01.731
     dvi:           Default installer: Enter 20:13:01.735
     dvi:                {Select Best Driver}
     dvi:                     Class GUID of device changed to: {4d36e97d-e325-11ce-
bfc1-08002be10318}.
     dvi:                     Selected Driver:
     dvi:                          Description - Intel(R) QM87 LPC Controller - 8C4F
     dvi:                          InfFile     -
c:\windows\system32\driverstore\filerepository\lynxpointsystem.inf_amd64_cd1e518d883
ecdfe\lynxpointsystem.inf
```

```
     dvi:                          Section     - Needs_ISAPNP_DRV
     dvi:                {Select Best Driver - exit(0x00000000)}
     dvi:           Default installer: Exit
     dvi:      {DIF_SELECTBESTCOMPATDRV - exit(0x00000000)} 20:13:01.743
     ndv:      Currently Installed Driver:
     ndv:           Inf Name       - oem1.inf
     ndv:           Driver Date    - 10/03/2016
     ndv:           Driver Version - 10.1.1.38
     ndv: {Update Device Driver - exit(00000103)}
!    ndv: No better matching drivers found for device
'PCI\VEN_8086&DEV_8C4F&SUBSYS_05BE1028&REV_04\3&11583659&0&F8'.
!    ndv: No devices were updated.
<<<  Section end 2019/09/26 20:13:01.759
<<<  [Exit status: FAILURE(0xC1900101)]
```

This analysis indicates that the Windows upgrade error can be resolved by deleting the `C:\ProgramData\Microsoft\Crypto\RSA\S-1-5-18\[CN]` file.

> **Note**
>
> In this example, the full file name is `C:\ProgramData\Microsoft\Crypto\RSA\S-1-5-18\be8228fb2d3cb6c6b0ccd9ad51b320b4_a43d512c-69f2-42de-aef9-7a88fabdaa3f`.

Related articles

- Fix Windows Update errors by using the DISM or System Update Readiness tool.

Last updated on 04/08/2025

Windows 10 upgrade resolution procedures

01/15/2025Applies to: Supported versions of Windows Client

Applies to: Windows 10

> **Note**
>
> This is a 200 level topic (moderate).
>
> For IT professionals, check more information in Resolve Windows 10 upgrade errors.

This article provides some common causes and solutions that are associated with specific upgrade error codes. If a Windows 10 upgrade fails, you can write down the error code that is displayed, or find the error code in the Windows Event Log or in the Windows Setup log files (ex: *setuperr.log*) and review the cause and solutions provided here. You should also try running the free SetupDiag tool provided by Microsoft, which can automatically find the reason for an upgrade failure.

0xC1900101

A frequently observed result code is 0xC1900101. This result code can be thrown at any stage of the upgrade process, with the exception of the downlevel phase. 0xC1900101 is a generic rollback code, and usually indicates that an incompatible driver is present. The incompatible driver can cause blue screens, system hangs, and unexpected reboots. Analysis of supplemental log files is often helpful, such as:

- The minidump file: *$Windows.~bt\Sources\Rollback\setupmem.dmp*
- Event logs: *$Windows.~bt\Sources\Rollback*.evtx*
- The device install log: *$Windows.~bt\Sources\Rollback\setupapi\setupapi.dev.log*

The device install log is helpful if rollback occurs during the sysprep operation (extend code 0x30018).

To resolve a rollback that was caused by driver conflicts, try running setup using a minimal set of drivers and startup programs by performing a clean boot before initiating the upgrade process.

See the following general troubleshooting procedures associated with a result code of 0xC1900101:

Expand table

Code	Mitigation	Cause
0xC1900101 - 0x20004	Uninstall antivirus applications. Remove all unused SATA devices. Remove all unused devices and drivers. Update drivers and BIOS.	Windows Setup encountered an error during the SAFE_OS with the INSTALL_RECOVERY_ENVIRONMENT operation. This error is caused by out-of-date drivers.
0xC1900101 - 0x2000c	Disconnect all peripheral devices that are connected to the system, except for the mouse, keyboard and display. Contact your hardware vendor to obtain updated device drivers.	Windows Setup encountered an unspecified error during Wim apply in the WinPE phase.

Code	Mitigation	Cause
	Ensure that "Download and install updates (recommended)" is accepted at the start of the upgrade process.	This error is caused by out-of-date drivers
0xC1900101 - 0x20017	Ensure that all that drivers are updated. Open the Setuperr.log and Setupact.log files in the %windir%\Panther directory, and then locate the problem drivers. For more information, see Windows Vista, Windows 7, Windows Server 2008 R2, Windows 8.1, and Windows 10 setup log file locations. Update or uninstall the problem drivers.	A driver has caused an illegal operation. Windows wasn't able to migrate the driver, resulting in a rollback of the operating system. This is a SafeOS boot failure, typically caused by drivers or non-Microsoft disk encryption software. This can also be caused by a hardware failure.
0xC1900101 - 0x30018	Disconnect all peripheral devices that are connected to the system, except for the mouse, keyboard and display. Contact your hardware vendor to obtain updated device drivers. Ensure that "Download and install updates (recommended)" is accepted at the start of the upgrade process.	A device driver has stopped responding to setup.exe during the upgrade process.
0xC1900101 - 0x3000D	Disconnect all peripheral devices that are connected to the system, except for the mouse, keyboard and display. Update or uninstall the display driver.	Installation failed during the FIRST_BOOT phase while attempting the MIGRATE_DATA operation. This can occur due to a problem with a display driver.
0xC1900101 - 0x4000D	Check supplemental rollback logs for a setupmem.dmp file, or event logs for any unexpected reboots or errors. Review the rollback log and determine the stop code. The rollback log is located in the *$Windows.~BT\Sources\Rollback* folder. An example analysis is shown below. This example isn't representative of all cases: Info SP Crash 0x0000007E detected Info SP Module name : Info SP Bugcheck parameter 1: 0xFFFFFFFFC0000005 Info SP Bugcheck parameter 2: 0xFFFFF8015BC0036A Info SP Bugcheck parameter 3: 0xFFFFD000E5D23728 Info SP Bugcheck parameter 4: 0xFFFFD000E5D22F40 Info SP Can't recover the system. Info SP Rollback: Showing splash window with restoring text: Restoring your previous version of Windows. Typically, there's a dump file for the crash to analyze. If you aren't equipped to debug the dump, then attempt the following basic troubleshooting procedures: 1. Make sure you have enough disk space. 2. If a driver is identified in the bug check message, disable the driver or check with the manufacturer for driver updates. 3. Try changing video adapters. 4. Check with your hardware vendor for any BIOS updates. 5. Disable BIOS memory options such as caching or shadowing.	A rollback occurred due to a driver configuration issue. Installation failed during the second boot phase while attempting the MIGRATE_DATA operation. This can occur because of incompatible drivers.

Code	Mitigation	Cause
0xC1900101 - 0x40017	Clean boot into Windows, and then attempt the upgrade to Windows 10. For more information, see How to perform a clean boot in Windows. Ensure that you select the option to "Download and install updates (recommended)." Computers that run Citrix VDA You may see this message after you upgrade a computer from Windows 10, version 1511 to Windows 10, version 1607. After the second system restart, the system generates this error and then rolls back to the previous version. This problem has also been observed in upgrades to Windows 8.1 and Windows 8. This problem occurs because the computer has Citrix Virtual Delivery Agent (VDA) installed. Citrix VDA installs device drivers and a file system filter driver (CtxMcsWbc). This Citrix filter driver prevents the upgrade from writing changes to the disk, so the upgrade can't complete and the system rolls back. Resolution To resolve this problem, install Cumulative update for Windows 10 Version 1607 and Windows Server 2016: November 8, 2016. You can work around this problem in two ways: Workaround 1 1. Use the VDA setup application (VDAWorkstationSetup_7.11) to uninstall Citrix VDA. 2. Run the Windows upgrade again. 3. Reinstall Citrix VDA. Workaround 2 If you can't uninstall Citrix VDA, follow these steps to work around this problem: 1. In Registry Editor, go to the following subkey: `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Class{4d36e967-e325-11ce-bfc1-08002be10318}\CtxMcsWbc` 2. Change the value of the `Start` entry from **0** to **4**. This change disables the Citrix MCS cache service. 3. Go to the following subkey: `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Class{4d36e967-e325-11ce-bfc1-08002be10318}` 4. Delete the `CtxMcsWbc` entry. 5. Restart the computer, and then try the upgrade again. Non-Microsoft information disclaimer The non-Microsoft products that this article discusses are manufactured by companies that are independent of Microsoft.	Windows 10 upgrade failed after the second reboot. This is caused by a faulty driver. For example: antivirus filter drivers or encryption drivers.

Code	Mitigation	Cause
	Microsoft makes no warranty, implied or otherwise, about the performance or reliability of these products.	

0x800xxxxx

Result codes that start with the digits 0x800 are also important to understand. These error codes indicate general operating system errors, and aren't unique to the Windows upgrade process. Examples include timeouts, devices not functioning, and a process stopping unexpectedly.

See the following general troubleshooting procedures associated with a result code of 0x800xxxxx:

Expand table

Code	Mitigation	Cause
80040005 - 0x20007	This error has more than one possible cause. Attempt quick fixes, and if not successful, analyze log files in order to determine the problem and solution.	An unspecified error occurred with a driver during the SafeOS phase.
0x80073BC3 - 0x20009 0x80070002 - 0x20009 0x80073B92 - 0x20009	These errors occur during partition analysis and validation, and can be caused by the presence of multiple system partitions. For example, if you installed a new system drive but left the previous system drive connected, this can cause a conflict. To resolve the errors, disconnect or temporarily disable drives that contain the unused system partition. You can reconnect the drive after the upgrade has completed. Alternatively, you can delete the unused system partition.	The requested system device can't be found, there's a sharing violation, or there are multiple devices matching the identification criteria.
800704B8 - 0x3001A	Disable or uninstall non-Microsoft antivirus applications, disconnect all unnecessary devices, and perform a clean boot.	An extended error has occurred during the first boot phase.
8007042B - 0x4000D	Analyze log files in order to determine the file, application, or driver that isn't able to be migrated. Disconnect, update, remove, or replace the device or object.	The installation failed during the second boot phase while attempting the MIGRATE_DATA operation. This issue can occur due to file system, application, or driver issues.
8007001F - 0x3000D	Analyze log files in order to determine the files or registry entries that are blocking data migration. This error can be due to a problem with user profiles. It can occur due to corrupt registry entries under `HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\ProfileList` or invalid files in the *\Users* directory. Note: If a previous upgrade didn't complete, invalid profiles might exist in the *Windows.old\Users* directory. To repair this error, ensure that deleted accounts aren't still present in the Windows registry and that files under the *\Users* directory are valid. Delete	The installation failed in the FIRST_BOOT phase with an error during MIGRATE_DATA operation.

Code	Mitigation	Cause
	the invalid files or user profiles that are causing this error. The specific files and profiles that are causing the error will be recorded in the Windows setup log files.	
8007001F - 0x4000D	Analyze log files in order to determine the device that isn't functioning properly. Disconnect, update, or replace the device.	General failure, a device attached to the system isn't functioning.
8007042B - 0x4001E	This error has more than one possible cause. Attempt quick fixes, and if not successful, analyze log files in order to determine the problem and solution.	The installation failed during the second boot phase while attempting the PRE_OOBE operation.

Other result codes

Expand table

Error code	Cause	Mitigation
0xC1800118	WSUS has downloaded content that it can't use due to a missing decryption key.	See Steps to resolve error 0xC1800118 for information.
0xC1900200	Setup.exe has detected that the machine doesn't meet the minimum system requirements.	Ensure the system you're trying to upgrade meets the minimum system requirements. See Windows 10 specifications for information.
0x80090011	A device driver error occurred during user data migration.	Contact your hardware vendor and get all the device drivers updated. It's recommended to have an active internet connection during upgrade process. Ensure that "Download and install updates (recommended)" is accepted at the start of the upgrade process.
0xC7700112	Failure to complete writing data to the system drive, possibly due to write access failure on the hard disk.	This issue is resolved in the latest version of Upgrade Assistant. Ensure that "Download and install updates (recommended)" is accepted at the start of the upgrade process.
0x80190001	An unexpected error was encountered while attempting to download files required for upgrade.	To resolve this issue, download and run the media creation tool. See Download windows 10.
0x80246007	The update wasn't downloaded successfully.	Attempt other methods of upgrading the operating system. Download and run the media creation tool. See Download windows 10. Attempt to upgrade using .ISO or USB. Note: Windows 10 Enterprise isn't available in the media creation tool. For more information, go to the Volume Licensing Service Center.

Error code	Cause	Mitigation
0x80244018	Your machine is connected through a proxy server.	Make sure Automatically Detect Settings is selected in internet options. (**Control Panel** > **Internet Options** > **Connections** > **LAN Settings**).
0xC1900201	The system didn't pass the minimum requirements to install the update.	Contact the hardware vendor to get the latest updates.
0x80240017	The upgrade is unavailable for this edition of Windows.	Administrative policies enforced by your organization might be preventing the upgrade. Contact your IT administrator.
0x80070020	The existing process can't access the file because it's being used by another process.	Use the MSCONFIG tool to perform a clean boot on the machine and then try to perform the update again. For more information, see How to perform a clean boot in Windows.
0x80070522	The user doesn't have required privilege or credentials to upgrade.	Ensure that you've signed in as a local administrator or have local administrator privileges.
0xC1900107	A cleanup operation from a previous installation attempt is still pending and a system reboot is required in order to continue the upgrade.	Restart the device and run setup again. If restarting the device doesn't resolve the issue, then use the Disk Cleanup utility to clean up the temporary files and the System files. For more information, see Disk cleanup in Windows 10.
0xC1900209	The user has chosen to cancel because the system doesn't pass the compatibility scan to install the update. Setup.exe will report this error when it can upgrade the machine with user data but cannot migrate installed applications.	Incompatible software is blocking the upgrade process. Uninstall the application and try the upgrade again. For more information, see Windows 10 Pre-Upgrade Validation using SETUP.EXE. You can also download the Windows Assessment and Deployment Kit (ADK) for Windows 10 and install Application Compatibility Tools.
0x8007002	This error is specific to upgrades using Configuration Manager R2 SP1 CU3 (5.00.8238.1403)	Analyze the SMSTS.log and verify that the upgrade is failing on "Apply Operating system" Phase: Error 80072efe DownloadFileWithRanges() failed. 80072efe. ApplyOperatingSystem (0x0760) The error 80072efe means that the connection with the server was terminated abnormally. To resolve this issue, try the OS Deployment test on a client in same VLAN as the Configuration Manager server. Check the network configuration for random client-server connection issues happening on the remote VLAN.

Error code	Cause	Mitigation
0x80240FFF	Occurs when update synchronization fails. It can occur when you're using Windows Server Update Services on its own or when it's integrated with Microsoft Endpoint Configuration Manager. If you enable update synchronization before you install hotfix 3095113, WSUS doesn't recognize the Upgrades classification, and instead treats the upgrade like a regular update.	You can prevent this by installing hotfix 3095113 before you enable update synchronization. However, if you have already run into this problem, do the following steps: 1. Disable the Upgrades classification. 2. Install hotfix 3095113. 3. Delete previously synched updates. 4. Enable the Upgrades classification. 5. Perform a full synch. For detailed information on how to run these steps check out How to delete upgrades in WSUS.
0x8007007E	Occurs when update synchronization fails because you don't have hotfix 3095113 installed before you enable update synchronization. Specifically, the CopyToCache operation fails on clients that have already downloaded the upgrade because Windows Server Update Services has bad metadata related to the upgrade. It can occur when you're using standalone Windows Server Update Services or when WSUS is integrated with Microsoft Endpoint Configuration Manager.	Use the following steps to repair Windows Server Update Services. You must run these steps on each WSUS server that synched metadata before you installed the hotfix. Stop the Windows Update service. • Sign in as a user with administrative privileges, and then do the following steps: • Open **Administrative Tools** from the **Control Panel**. • Double-click **Services**. • Find the **Windows Update** service, right-click it, and then select **Stop**. If prompted, enter your credentials. Delete all files and folders under *c:\Windows\SoftwareDistribution\DataStore*. Restart the Windows Update service.

Other error codes

Expand table

Error Codes	Cause	Mitigation
0x80070003-0x20007	This is a failure during SafeOS phase driver installation.	Verify device drivers on the computer, and analyze log files to determine the problem driver.
0x8007025D - 0x2000C	This error occurs if the ISO file's metadata is corrupt or if there's an issue with the storage medium, such as a RAM module containing bad blocks during the installation of Windows.	Redownload the ISO/Media and reattempt the upgrade Alternatively, re-create installation media the Media Creation Tool.
0x80070490 - 0x20007	An incompatible device driver is present.	Verify device drivers on the computer, and analyze log files to determine the problem driver.
0xC1900101 - 0x2000c	An unspecified error occurred in the SafeOS phase during WIM apply. This can be caused by an	Run checkdisk to repair the file system. For more information, see the quick fixes section in this guide. Update drivers on the computer, and select "Download and install

Error Codes	Cause	Mitigation
	outdated driver or disk corruption.	updates (recommended)" during the upgrade process. Disconnect devices other than the mouse, keyboard and display.
0xC1900200 - 0x20008	The computer doesn't meet the minimum requirements to download or upgrade to Windows 10.	See Windows 10 Specifications and verify the computer meets minimum requirements. Review logs for compatibility information.
0xC1900200 - 0x20008	The computer doesn't meet the minimum requirements to download or upgrade to Windows 10. See Windows 10 Specifications and verify the computer meets minimum requirements. Review logs for Windows 10 Specifications .	
0x80070004 - 0x3000D	This is a problem with data migration during the first boot phase. There are multiple possible causes.	Analyze log files to determine the issue.
0xC1900101 - 0x4001E	Installation failed in the SECOND_BOOT phase with an error during PRE_OOBE operation.	This is a generic error that occurs during the OOBE phase of setup. See the 0xC1900101 section of this guide and review general troubleshooting procedures described in that section.
0x80070005 - 0x4000D	The installation failed in the SECOND_BOOT phase with an error in during MIGRATE_DATA operation. This error indicates that access was denied while attempting to migrate data.	Analyze log files to determine the data point that is reporting access denied.
0x80070004 - 0x50012	Windows Setup failed to open a file.	Analyze log files to determine the data point that is reporting access problems.
0xC190020e 0x80070070 - 0x50011 0x80070070 - 0x50012 0x80070070 - 0x60000	These errors indicate the computer doesn't have enough free space available to install the upgrade.	To upgrade a computer to Windows 10, it requires 16 GB of free hard drive space for a 32-bit OS, and 20 GB for a 64-bit OS. If there isn't enough space, attempt to free up drive space before proceeding with the upgrade. **Note:** If your device allows it, you can use an external USB drive for the upgrade process. Windows setup will back up the previous version of Windows to a USB external drive. The external drive must be at least 8 GB (16 GB is recommended). The external drive should be formatted using NTFS. Drives that are formatted in FAT32 may run into errors due to FAT32 file size limitations. USB drives are preferred over SD cards because drivers for SD cards aren't migrated if the device doesn't support Connected Standby.

Modern setup errors

Also see the following sequential list of modern setup (mosetup) error codes with a brief description of the cause.

Expand table

Result code	Message	Description
0XC1900100	MOSETUP_E_VERSION_MISMATCH	An unexpected version of Setup Platform binaries was encountered. Verify the package contents.
0XC1900101	MOSETUP_E_SETUP_PLATFORM	The Setup Platform has encountered an unspecified error.
0XC1900102	MOSETUP_E_SHUTDOWN_BLOCK	Unable to create or destroy the shutdown block message.
0XC1900103	MOSETUP_E_COMPAT_TIMEOUT	The compatibility issues weren't resolved within the required time limit.
0XC1900104	MOSETUP_E_PROCESS_TIMEOUT	The installation process did not complete within the required time limit.
0XC1900105	MOSETUP_E_TEST_MODE	The installation process is being used in a test environment.
0XC1900106	MOSETUP_E_TERMINATE_PROCESS	The installation process was terminated.
0XC1900107	MOSETUP_E_CLEANUP_PENDING	A cleanup operation from a previous installation attempt is still pending. A system reboot is required.
0XC1900108	MOSETUP_E_REPORTING	An error has occurred and the result value must be consolidated for telemetry purposes.
0XC1900109	MOSETUP_E_COMPAT_TERMINATE	The installation process was terminated during the actionable compatibility phase.
0XC190010a	MOSETUP_E_UNKNOWN_CMD_LINE	The installation process was launched with an unknown command-line argument.
0XC190010b	MOSETUP_E_INSTALL_IMAGE_NOT_FOUND	The installation image was not found.
0XC190010c	MOSETUP_E_AUTOMATION_INVALID	The provided automation information was invalid.
0XC190010d	MOSETUP_E_INVALID_CMD_LINE	The installation process was launched with an invalid command-line argument.

Result code	Message	Description
0XC190010e	MOSETUP_E_EULA_ACCEPT_REQUIRED	The installation process requires that the user accept the license agreement.
0XC1900110	MOSETUP_E_EULA_CANCEL	The user has chosen to cancel for license agreement.
0XC1900111	MOSETUP_E_ADVERTISE_CANCEL	The user has chosen to cancel for advertisement.
0XC1900112	MOSETUP_E_TARGET_DRIVE_NOT_FOUND	Could not find a target drive letter.
0XC1900113	MOSETUP_E_EULA_DECLINED	The user has declined the license terms.
0XC190011e	MOSETUP_E_FLIGHTING_BVT	The installation process has been halted for testing purposes.
0XC190011f	MOSETUP_E_PROCESS_CRASHED	The installation process crashed.
0XC1900120	MOSETUP_E_EULA_TIMEOUT	The user has not accepted the EULA within the required time limit.
0XC1900121	MOSETUP_E_ADVERTISE_TIMEOUT	The user has not accepted Advertisement within the required time limit.
0XC1900122	MOSETUP_E_DOWNLOADDISKSPACE_TIMEOUT	The download disk space issues were not resolved within the required time limit.
0XC1900123	MOSETUP_E_INSTALLDISKSPACE_TIMEOUT	The install disk space issues were not resolved within the required time limit.
0XC1900124	MOSETUP_E_COMPAT_SYSREQ_TIMEOUT	The minimum requirements compatibility issues were not resolved within the required time limit.
0XC1900125	MOSETUP_E_COMPAT_DOWNLOADREQ_TIMEOUT	The compatibility issues for download were not resolved within the required time limit.
0XC1900126	MOSETUP_E_GATHER_OS_STATE_SIGNATURE	The GatherOsState executable has invalid signature.
0XC1900127	MOSETUP_E_UNINSTALL_ALLOWED_ABORT	The user has chosen to abort Setup to keep Uninstall option active.
0XC1900128	MOSETUP_E_MISSING_TASK	The install cannot continue because a required task is missing.
0XC1900129	MOSETUP_E_UPDATEMEDIA_REQUESTED	A more up-to-date version of setup will be launched to continue installation

Result code	Message	Description
0XC190012f	MOSETUP_E_FINALIZE_ALREADY_REQUESTED	The install cannot continue because a finalize operation was already requested.
0XC1900130	MOSETUP_E_INSTALL_HASH_MISSING	The install cannot continue because the instance hash was not found.
0XC1900131	MOSETUP_E_INSTALL_HASH_MISMATCH	The install cannot continue because the instance hash does not match.
0XC19001df	MOSETUP_E_DISK_FULL	The install cannot continue because the system is out of disk space.
0XC19001e0	MOSETUP_E_GATHER_OS_STATE_FAILED	The GatherOsState executable has failed to execute.
0XC19001e1	MOSETUP_E_PROCESS_SUSPENDED	The installation process was suspended.
0XC19001e2	MOSETUP_E_PREINSTALL_SCRIPT_FAILED	A preinstall script failed to execute or returned an error.
0XC19001e3	MOSETUP_E_PRECOMMIT_SCRIPT_FAILED	A precommit script failed to execute or returned an error.
0XC19001e4	MOSETUP_E_FAILURE_SCRIPT_FAILED	A failure script failed to execute or returned an error.
0XC19001e5	MOSETUP_E_SCRIPT_TIMEOUT	A script exceeded the timeout limit.
0XC1900200	MOSETUP_E_COMPAT_SYSREQ_BLOCK	The system does not pass the minimum requirements to install the update.
0XC1900201	MOSETUP_E_COMPAT_SYSREQ_CANCEL	The user has chosen to cancel because the system does not pass the minimum requirements to install the update.
0XC1900202	MOSETUP_E_COMPAT_DOWNLOADREQ_BLOCK	The system does not pass the minimum requirements to download the update.
0XC1900203	MOSETUP_E_COMPAT_DOWNLOADREQ_CANCEL	The user has chosen to cancel because the system does not pass the minimum requirements to download the update.
0XC1900204	MOSETUP_E_COMPAT_MIGCHOICE_BLOCK	The system does not pass the requirements for desired migration choice.
0XC1900205	MOSETUP_E_COMPAT_MIGCHOICE_CANCEL	The user has chosen to cancel because the system does not pass

Result code	Message	Description
		the requirements for desired migration choice.
0XC1900206	MOSETUP_E_COMPAT_DEVICEREQ_BLOCK	The system does not pass the device scan to install the update.
0XC1900207	MOSETUP_E_COMPAT_DEVICEREQ_CANCEL	The user has chosen to cancel because the system does not pass the device scan to install the update.
0XC1900208	MOSETUP_E_COMPAT_INSTALLREQ_BLOCK	The system does not pass the compat scan to install the update.
0XC1900209	MOSETUP_E_COMPAT_INSTALLREQ_CANCEL	The user has chosen to cancel because the system does not pass the compat scan to install the update.
0XC190020a	MOSETUP_E_COMPAT_RECOVERYREQ_BLOCK	The system does not pass the minimum requirements to recover Windows.
0XC190020b	MOSETUP_E_COMPAT_RECOVERYREQ_CANCEL	The user has chosen to cancel because the system does not pass the minimum requirements to recover Windows.
0XC190020c	MOSETUP_E_DOWNLOADDISKSPACE_BLOCK	The system does not pass the disk space requirements to download the payload.
0XC190020d	MOSETUP_E_DOWNLOADDISKSPACE_CANCEL	The user has chosen to cancel as the device does not have enough disk space to download.
0XC190020e	MOSETUP_E_INSTALLDISKSPACE_BLOCK	The system does not pass the disk space requirements to install the payload.
0XC190020f	MOSETUP_E_INSTALLDISKSPACE_CANCEL	The user has chosen to cancel as the device does not have enough disk space to install.
0XC1900210	MOSETUP_E_COMPAT_SCANONLY	The user has used the setup.exe command line to do scanonly, not to install the OS.
0XC1900211	MOSETUP_E_DOWNLOAD_UNPACK_DISKSPACE_BLOCK	The system does not pass the disk space requirements to download and unpack media.
0XC1900212	MOSETUP_E_DOWNLOAD_UNPACK_DISKSPACE_MULTIARCH_BLOCK	The system does not pass the disk space requirements to download and unpack multi-architecture media.

Result code	Message	Description
0XC1900213	MOSETUP_E_NO_OFFER_FOUND	There was no offer found that matches the required criteria.
0XC1900214	MOSETUP_E_UNSUPPORTED_VERSION	This version of the tool is not supported.
0XC1900215	MOSETUP_E_NO_MATCHING_INSTALL_IMAGE	Could not find an install image for this system.
0XC1900216	MOSETUP_E_ROLLBACK_PENDING	Found pending OS rollback operation.
0XC1900220	MOSETUP_E_COMPAT_REPORT_NOT_DISPLAYED	The compatibility report cannot be displayed due to a missing system component.
0XC1900400	MOSETUP_E_UA_VERSION_MISMATCH	An unexpected version of Update Agent client was encountered.
0XC1900401	MOSETUP_E_UA_NO_PACKAGES_TO_DOWNLOAD	No packages to be downloaded.
0XC1900402	MOSETUP_E_UA_UPDATE_CANNOT_BE_MERGED	No packages to be downloaded.
0XC1900403	MOSETUP_E_UA_CORRUPT_PAYLOAD_FILES	Payload files were corrupt.
0XC1900404	MOSETUP_E_UA_BOX_NOT_FOUND	The installation executable was not found.
0XC1900405	MOSETUP_E_UA_BOX_CRASHED	The installation process terminated unexpectedly.

Data collection

If you need assistance from Microsoft support, we recommend you collect the information by following the steps mentioned in Gather information by using TSS for deployment-related issues.

More information

- Windows 10 FAQ for IT professionals
- Windows 10 Enterprise system requirements
- Windows 10 Specifications
- Windows 10 IT pro forums
- Fix Windows Update errors by using the DISM or System Update Readiness tool
- Windows 7 to Windows 10 upgrade error (0x800707E7 - 0x3000D))
- Windows 10 upgrade error: User profile suffix mismatch, 0x800707E7 - 0x3000D

Submit Windows upgrade errors using Feedback Hub

Applies to: ✅ Windows 11, ✅ Windows 10

> ⓘ **Note**
>
> This article is a 100 level article (basic).
>
> See Resolve Windows upgrade errors for a full list of articles in this section.

This article describes how to submit problems with a Windows upgrade to Microsoft using the Windows Feedback Hub.

About the Feedback Hub

The Feedback Hub app allows reporting to Microsoft of any problems encountered while using Windows. It also allows sending suggestions to Microsoft on how to improve the Windows experience. Previously, the Feedback Hub could only be used through the Windows Insider Program. Now anyone can use this tool. The Feedback Hub app can be downloaded from the Microsoft Store.

The Feedback Hub requires a currently supported version of Windows. The Feedback Hub can be used to submit information to Microsoft if problems are encountered while upgrading Windows. If upgrading between currently supported version of Windows, the Feedback Hub automatically collects log files. For out of support operating systems before Windows 10 that don't support the Feedback Hub, the log files must be manually collected. The log files can then be attached to the feedback item using a device that is running a currently supported version of Windows that supports the Feedback Hub.

Submit feedback

To submit feedback about a failed Windows upgrade, open the Feedback Hub.

In the Feedback Hub, fill out all of the sections with as much detail as possible:

1. **Enter your feedback**
2. **Choose a category**
3. **Find similar feedback** - this section doesn't have anything to fill out, but it is important to check for similar feedback items. If a similar feedback item is found, select it and then

select the **Next** button. This allows Microsoft to see how many people are affected by the same issue.

4. **Add more details**

Recommended information that can be included under the **Add more details** section include:

- When did the failure occur?
  - Were there any reboots?
  - How many times did the system reboot?
- How did the upgrade fail?
  - Were any error codes visible?
  - Did the computer fail to a blue screen?
  - Did the computer automatically rollback or did it hang, requiring the computer to be power cycled before it rolled back?
- What type of security software is installed?
- Is the computer up to date with latest drivers and firmware?
- Are there any external devices connected?

Using the **Attach a screenshot** and **Attach a file** options allows screenshots or files to be included as part of the feedback item. Attachments and screenshots are optional, but can be helpful when diagnosing the upgrade issue. For example, log files can be included as attachments to the feedback item. The location of the Windows upgrade log files is described in the article Windows Setup log files and event logs.

Finally the **Recreate my problem** option can be used to potentially send additional data and logs for Microsoft to evaluate.

Once all the feedback items are completed, select the **Submit** button to send the feedback. Microsoft receives the feedback and begins analyzing the issue. The submitted feedback can be checked on periodically to see what solutions are provided.

Link to the feedback

After the feedback is submitted, additional information and items can be added to the feedback item. To do so:

1. Open the Feedback Hub.
2. At the top of the Feedback Hub, select **My feedback**.
3. Select the feedback item that was submitted.
4. Select **Share**.
5. Copy and then use the short link that is displayed.

Short Link https://aka.ms/OI70d7 Copy link Share

Last updated on 04/08/2025

Windows Update issues troubleshooting

04/25/2025Applies to: Supported versions of Windows Client

Applies to: Windows 10

Try our Virtual Agent - It can help you quickly identify and fix common Windows Update issues

If you run into problems when using Windows Update, start with the following steps:

1. Run the built-in Windows Update troubleshooter to fix common issues. Navigate to **Settings** > **Update & Security** > **Troubleshoot** > **Windows Update**.
2. Install the most recent Servicing Stack Update that matches your version of Windows from the Microsoft Update Catalog. For more information on servicing stack updates, see Servicing stack updates.
3. Make sure that you install the latest Windows updates, cumulative updates, and rollup updates. To verify the update status, refer to the appropriate update history for your system:
   - Windows 10, version 2004 and Windows Server, version 2004
   - Windows 10, version 1909 and Windows Server, version 1909
   - Windows 10, version 1903 and Windows Server, version 1903
   - Windows 10, version 1809 and Windows Server 2019
   - Windows 10, version 1803
   - Windows 10, version 1709
   - Windows 10, version 1703
   - Windows 10 and Windows Server 2016
   - Windows 8.1 and Windows Server 2012 R2
   - Windows Server 2012
   - Windows 7 SP1 and Windows Server 2008 R2 SP1

Advanced users can also refer to the log generated by Windows Update for further investigation.

You might encounter the following scenarios when using Windows Update.

Why am I offered an older update?

The update that is offered to a device depends on several factors. The following are some of the most common attributes:

- OS Build
- OS Branch
- OS Locale
- OS Architecture
- Device update management configuration

If the update you're offered isn't the most current available, it might be because your device is being managed by a WSUS server, and you're being offered the updates available on that server. It's also possible, if your device is part of a deployment group, that your admin is intentionally slowing the rollout of updates. Since the deployment is slow and measured to begin with, all devices won't receive the update on the same day.

My device is frozen at scan. Why?

The Settings UI communicates with the Update Orchestrator service that in turn communicates with to Windows Update service. If these services stop unexpectedly, then you might see this behavior. In such cases, follow these steps:

1. Close the Settings app and reopen it.

2. Start Services.msc and check if the following services are running:

    - Update State Orchestrator
    - Windows Update

Feature updates aren't being offered while other updates are

Devices running Windows 10, version 1709 through Windows 10, version 1803 that are configured to update from Windows Update (including Windows Update for Business) are able to install servicing and definition updates but are never offered feature updates.

Checking the WindowsUpdate.log reveals the following error:

Output

```
YYYY/MM/DD HH:mm:ss:SSS PID  TID  Agent           * START * Finding updates
CallerId = Update;taskhostw  Id = 25
YYYY/MM/DD HH:mm:ss:SSS PID  TID  Agent           Online = Yes; Interactive = No;
AllowCachedResults = No; Ignore download priority = No
```

```
YYYY/MM/DD HH:mm:ss:SSS PID  TID  Agent           ServiceID = {855E8A7C-ECB4-4CA3-
B045-1DFA50104289} Third party service
YYYY/MM/DD HH:mm:ss:SSS PID  TID  Agent           Search Scope = {Current User}
YYYY/MM/DD HH:mm:ss:SSS PID  TID  Agent           Caller SID for Applicability: S-
1-12-1-2933642503-1247987907-1399130510-4207851353
YYYY/MM/DD HH:mm:ss:SSS PID  TID  Misc            Got 855E8A7C-ECB4-4CA3-B045-
1DFA50104289 redir Client/Server URL:
https://fe3.delivery.mp.microsoft.com/ClientWebService/client.asmx""
YYYY/MM/DD HH:mm:ss:SSS PID  TID  Misc            Token Requested with 0 category
IDs.
YYYY/MM/DD HH:mm:ss:SSS PID  TID  Misc            GetUserTickets: No user tickets
found. Returning WU_E_NO_USERTOKEN.
YYYY/MM/DD HH:mm:ss:SSS PID  TID  Misc            *FAILED* [80070426] Method
failed [AuthTicketHelper::GetDeviceTickets:570]
YYYY/MM/DD HH:mm:ss:SSS PID  TID  Misc            *FAILED* [80070426] Method
failed [AuthTicketHelper::GetDeviceTickets:570]
YYYY/MM/DD HH:mm:ss:SSS PID  TID  Misc            *FAILED* [80070426]
GetDeviceTickets
YYYY/MM/DD HH:mm:ss:SSS PID  TID  Misc            *FAILED* [80070426] Method
failed [AuthTicketHelper::AddTickets:1092]
YYYY/MM/DD HH:mm:ss:SSS PID  TID  Misc            *FAILED* [80070426] Method
failed [CUpdateEndpointProvider::GenerateSecurityTokenWithAuthTickets:1587]
YYYY/MM/DD HH:mm:ss:SSS PID  TID  Misc            *FAILED* [80070426]
GetAgentTokenFromServer
YYYY/MM/DD HH:mm:ss:SSS PID  TID  Misc            *FAILED* [80070426]
GetAgentToken
YYYY/MM/DD HH:mm:ss:SSS PID  TID  Misc            *FAILED* [80070426] EP:Call to
GetEndpointToken
YYYY/MM/DD HH:mm:ss:SSS PID  TID  Misc            *FAILED* [80070426] Failed to
obtain service 855E8A7C-ECB4-4CA3-B045-1DFA50104289 plugin Client/Server auth
token of type 0x00000001
YYYY/MM/DD HH:mm:ss:SSS PID  TID  ProtocolTalker  *FAILED* [80070426] Method
failed [CAgentProtocolTalkerContext::DetermineServiceEndpoint:377]
YYYY/MM/DD HH:mm:ss:SSS PID  TID  ProtocolTalker  *FAILED* [80070426]
Initialization failed for Protocol Talker Context
YYYY/MM/DD HH:mm:ss:SSS PID  TID  Agent           Exit code = 0x80070426
YYYY/MM/DD HH:mm:ss:SSS PID  TID  Agent           * END * Finding updates CallerId
= Update;taskhostw  Id = 25
```

The 0x80070426 error code translates to:

Output

```
ERROR_SERVICE_NOT_ACTIVE - # The service has not been started.
```

Microsoft Account Sign In Assistant (MSA or wlidsvc) is the service in question. The DCAT Flighting service (ServiceId: 855E8A7C-ECB4-4CA3-B045-1DFA50104289) relies on MSA to get the global device ID for the device. Without the MSA service running, the global device ID won't be generated and sent by the client and the search for feature updates never completes successfully.

To resolve this issue, reset the MSA service to the default StartType of "manual."

Issues related to HTTP/Proxy

Windows Update uses WinHttp with Partial Range requests (RFC 7233) to download updates and applications from Windows Update servers or on-premises WSUS servers. Therefore proxy servers on the network must support HTTP RANGE requests. If a proxy was configured in Internet Explorer (User level) but not in WinHTTP (System level), connections to Windows Update will fail.

To fix this issue, configure a proxy in WinHTTP by using the following netsh command:

Console

```
netsh winhttp set proxy ProxyServerName:PortNumber
```

> **Note**
>
> You can also import the proxy settings from Internet Explorer by using the following command: `netsh winhttp import proxy source=ie`.

If downloads through a proxy server fail with a 0x80d05001 DO_E_HTTP_BLOCKSIZE_MISMATCH error, or if you notice high CPU usage while updates are downloading, check the proxy configuration to permit HTTP RANGE requests to run.

You might choose to apply a rule to permit HTTP RANGE requests for the following URLs:

- `*.download.windowsupdate.com`
- `*.dl.delivery.mp.microsoft.com`
- `*.delivery.mp.microsoft.com`

If you can't allow RANGE requests, you'll be downloading more content than needed in updates (as delta patching won't work).

The update isn't applicable to your computer

The most common reasons for this error are described in the following table:

Expand table

Cause	Explanation	Resolution
Update is superseded	As updates for a component are released, the updated component will supersede an older component that is already on the system. When this issue occurs, the previous update is marked as superseded. If the update that you're trying to install already has a newer version of the payload on your system, you might receive this error message.	Check that the package that you're installing contains newer versions of the binaries. Or, check that the package is superseded by another new package.
Update is already installed	If the update that you're trying to install was previously installed, for example, by another update that carried the same payload, you may encounter this error message.	Verify that the package that you're trying to install wasn't previously installed.
Wrong update for architecture	Updates are published by CPU architecture. If the update that you're trying to install doesn't match the architecture for your CPU, you may encounter this error message.	Verify that the package that you're trying to install matches the Windows version that you're using. The Windows version information can be found in the "Applies To" section of the article for each update. For example, Windows Server 2012-only updates can't be installed on Windows Server 2012 R2-based computers. Also, verify that the package that you're installing matches the processor architecture of the Windows version that you're using. For example, an x86-based update can't be installed on x64-based installations of Windows.
Missing prerequisite update	Some updates require a prerequisite update before they can be applied to a system. If you're missing a prerequisite update, you may encounter this error message. For example, KB 2919355 must be installed on Windows 8.1 and Windows Server 2012 R2 computers before many of the updates that were released after April 2014 can be installed.	Check the related articles about the package in the Microsoft Knowledge Base (KB) to make sure that you have the prerequisite updates installed. For example, if you encounter the error message on Windows 8.1 or Windows Server 2012 R2, you may have to install the April 2014 update 2919355 as a prerequisite and one or more pre-requisite servicing updates (KB 2919442 and KB 3173424). To determine if these prerequisite updates are installed, run the following PowerShell command: `get-hotfix KB3173424,KB2919355, KB2919442`. If the updates are installed, the command will return the installed date in the `InstalledOn` section of the output.

Error 0x800706D9 if the Windows Defender Firewall service is disabled

You observe that the download for updates is stuck at 0%, especially in Windows Server 2016, where the Background Intelligent Transfer Service (BITS) service is the default Download Manager.

Error that you might see in Windows Update logs:

Output

```
DownloadManager    Error 0x800706d9 occurred while downloading update; notifying
dependent calls.
```

Or

Output

```
[DownloadManager] BITS job {A4AC06DD-D6E6-4420-8720-7407734FDAF2} hit a transient
error, updateId = {D053C08A-6250-4C43-A111-56C5198FE142}.200 <NULL>, error =
0x800706D9
```

Or

Output

```
DownloadManager [0]12F4.1FE8::09/29/2017-13:45:08.530 [agent]DO job {C6E2F6DC-
5B78-4608-B6F1-0678C23614BD} hit a transient error, updateId = 5537BD35-BB74-40B2-
A8C3-B696D3C97CBA.201 <NULL>, error = 0x80D0000A
```

Go to **Services.msc** and ensure that the **Windows Defender Firewall** service is enabled. Stopping the service associated with Windows Firewall with Advanced Security isn't supported by Microsoft. For more information, see I need to disable Windows Firewall.

Issues arising from configuration of conflicting policies

Windows Update provides a wide range configuration policy to control the behavior of the Windows Update service in a managed environment. While these policies let you configure the settings at a granular level, misconfiguration or setting conflicting policies may lead to unexpected behaviors.

For more information, see How to configure automatic updates by using Group Policy or registry settings.

Device can't access update files

Ensure that devices can reach necessary Windows Update endpoints through the firewall. For example, for Windows 10, version 2004, the following protocols must be able to reach these respective endpoints:

Expand table

Protocol	Endpoint URL
TLS 1.2	`*.prod.do.dsp.mp.microsoft.com`
HTTP	`emdl.ws.microsoft.com`
HTTP	`*.dl.delivery.mp.microsoft.com`
HTTP	`*.windowsupdate.com`
HTTPS	`*.delivery.mp.microsoft.com`
TLS 1.2	`*.update.microsoft.com`
TLS 1.2	`tsfe.trafficshaping.dsp.mp.microsoft.com`

> **Note**
>
> Be sure not to use HTTPS for those endpoints that specify HTTP, and vice versa. The connection will fail.

The specific endpoints can vary between Windows client versions. See, for example, Windows 10 2004 Enterprise connection endpoints. Similar articles for other Windows client versions are available in the table of contents nearby.

Updates aren't downloading from the intranet endpoint (WSUS or Configuration Manager)

Windows client devices can receive updates from various sources, including Windows Update online, a Windows Server Update Services server, and others. To determine the source of Windows Updates currently being used on a device, follow these steps:

1. Start Windows PowerShell as an administrator.

2. Run the cmdlet:

```
PowerShell
$MUSM = New-Object -ComObject "Microsoft.Update.ServiceManager"
```

3. Run the cmdlet:

```
PowerShell
$MUSM.Services
```

Check the output for the Name and OffersWindowsUPdates parameters, which you can interpret according to this table.

Expand table

Output	Meaning
- Name: Microsoft Update -OffersWindowsUpdates: True	- The update source is Microsoft Update, which means that updates for other Microsoft products besides the operating system could also be delivered. - Indicates that the client is configured to receive updates for all Microsoft Products (Office, etc.)
- Name: DCat Flighting Prod - OffersWindowsUpdates: True	- Starting with Windows 10, version 1709, feature updates are always delivered through the DCAT service. - Indicates that the client is configured to receive feature updates from Windows Update.
- Name: Windows Store (DCat Prod) - OffersWindowsUpdates: False	-The update source is Insider Updates for Store Apps. - Indicates that the client won't receive or isn't configured to receive these updates.
- Name: Windows Server Update Service - OffersWindowsUpdates: True	- The source is a Windows Server Updates Services server. - The client is configured to receive updates from WSUS.
- Name: Windows Update - OffersWindowsUpdates: True	- The source is Windows Update. - The client is configured to receive updates from Windows Update Online.

You have a bad setup in the environment

In this example, per the Group Policy set through registry, the system is configured to use WSUS to download updates (note the second line):

Console

```
HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\WindowsUpdate\AU]
"UseWUServer"=dword:00000001
```

From Windows Update logs:

Output

```
2018-08-06 09:33:31:085  480 1118 Agent ** START **  Agent: Finding updates
[CallerId = OperationalInsight  Id = 49]
2018-08-06 09:33:31:085  480 1118 Agent *********
2018-08-06 09:33:31:085  480 1118 Agent   * Include potentially superseded updates
2018-08-06 09:33:31:085  480 1118 Agent   * Online = No; Ignore download priority
= No
2018-08-06 09:33:31:085  480 1118 Agent   * Criteria = "IsHidden = 0 AND
DeploymentAction=*"
2018-08-06 09:33:31:085  480 1118 Agent   * ServiceID = {00000000-0000-0000-0000-
000000000000} Third party service
2018-08-06 09:33:31:085  480 1118 Agent   * Search Scope = {Machine}
2018-08-06 09:33:32:554  480 1118 Agent   * Found 83 updates and 83 categories in
search; evaluated appl. rules of 517 out of 1473 deployed entities
2018-08-06 09:33:32:554  480 1118 Agent *********
2018-08-06 09:33:32:554  480 1118 Agent **  END  **  Agent: Finding updates
[CallerId = OperationalInsight  Id = 49]
```

In the above log snippet, we see that the `Criteria = "IsHidden = 0 AND DeploymentAction=*"`. "*" means there is nothing specified from the server. So, the scan happens but there is no direction to download or install to the agent. So it just scans the update and provides the results.

As shown in the following logs, automatic update runs the scan and finds no update approved for it. So it reports there are no updates to install or download. This is due to an incorrect configuration. The WSUS side should approve the updates for Windows Update so that it fetches the updates and installs them at the specified time according to the policy. Since this scenario doesn't include Configuration Manager, there's no way to install unapproved updates. You're expecting the operational insight agent to do the scan and automatically trigger the download and installation but that won't happen with this configuration.

Output

```
2018-08-06 10:58:45:992  480 5d8 Agent ** START **  Agent: Finding updates
[CallerId = AutomaticUpdates  Id = 57]
2018-08-06 10:58:45:992  480 5d8 Agent *********
```

```
2018-08-06 10:58:45:992  480 5d8 Agent   * Online = Yes; Ignore download priority
= No
2018-08-06 10:58:45:992  480 5d8 Agent   * Criteria = "IsInstalled=0 and
DeploymentAction='Installation' or IsPresent=1 and
DeploymentAction='Uninstallation' or IsInstalled=1 and
DeploymentAction='Installation' and RebootRequired=1 or IsInstalled=0 and
DeploymentAction='Uninstallation' and RebootRequired=1"

2018-08-06 10:58:46:617  480 5d8 PT   + SyncUpdates round trips: 2
2018-08-06 10:58:47:383  480 5d8 Agent   * Found 0 updates and 83 categories in
search; evaluated appl. rules of 617 out of 1473 deployed entities
2018-08-06 10:58:47:383  480 5d8 Agent Reporting status event with 0 installable,
83 installed,  0 installed pending, 0 failed and 0 downloaded updates
2018-08-06 10:58:47:383  480 5d8 Agent *********
2018-08-06 10:58:47:383  480 5d8 Agent **  END  **  Agent: Finding updates
[CallerId = AutomaticUpdates  Id = 57]
```

High bandwidth usage on Windows client by Windows Update

Users might see that Windows is consuming all the bandwidth in the different offices under the system context. This behavior is by design. Components that might consume bandwidth expand beyond Windows Update components.

The following group policies can help mitigate this situation:

- Blocking access to Windows Update servers: Policy Turn off access to all Windows Update features (Set to enabled)
- Driver search: Policy Specify search order for device driver source locations (Set to "Do not search Windows Update")
- Windows Store automatic update: Policy Turn off Automatic Download and Install of updates (Set to enabled)

Other components that connect to the internet:

- Windows Spotlight: Policy Configure Windows spotlight on lock screen (Set to disabled)
- Consumer experiences: Policy Turn off Microsoft consumer experiences (Set to enabled)
- Background traffic from Windows apps: Policy Let Windows apps run in the background

Transient errors caused by heavy load or network congestion

Users might receive the following errors from Windows Update. These errors are transient errors, occurring when the service is temporarily under heavy load or when networks are

congested. Users don't need to take any action because the device will retry the operation later.

Expand table

Error code	Error value	Details
WU_S_SEARCH_LOAD_SHEDDING	0x248001	Search operation completed successfully but one or more services were shedding load.
WU_E_PT_LOAD_SHEDDING	0x8024402d	The server is shedding load.

In these cases, users that programmatically call into the Windows Update Agent API to retrieve the result of a search operation would get orcFailed or orcSucceededWithErrors. Retrying the operation later is expected to succeed.

Data collection

If you need assistance from Microsoft support, we recommend you collect the information by following the steps mentioned in Gather information by using TSS for deployment-related issues.

Windows Update common errors and mitigation

01/15/2025Applies to: Supported versions of Windows Client

Try our Virtual Agent - It can help you quickly identify and fix common Windows Update issues

The following table provides information about common errors you might run into with Windows Update, and gives the steps to help you mitigate them.

Applies to: Windows 10, Windows 11

0x8024402F

Expand table

Message	Description	Mitigation
WU_E_PT_ECP_SUCCEEDED_WITH_ERRORS	External .cab file processing completed with some errors	This error can be caused by the Lightspeed Rocket for web filtering software. Add the IP addresses of devices you want to get updates to the exceptions list of Lightspeed Rocket.

0x80242006

Expand table

Message	Description	Mitigation

WU_E_UH_INVALIDMETADATA	A handler operation couldn't be completed because the update contains invalid metadata.	Rename the software redistribution folder and try to download the updates again:
		Rename the following folders to *.BAK:
		-%systemroot%\system32\catroot2
		
		Type the following commands at a command prompt. Press ENTER after you type each command.
		`Ren %systemroot%\SoftwareDistribution\DataStore DataStore.bak`
		`Ren %systemroot%\SoftwareDistribution\Download Download.bak`
		`Ren %systemroot%\system32\catroot2 catroot2.bak`

0x80070BC9

Expand table

Message	Description	Mitigation
ERROR_FAIL_REBOOT_REQUIRED	The requested operation failed. Restart the system to roll back changes made.	Ensure that you don't have any policies that control the start behavior of the Windows Installer service. This service should be managed by the operating system. The default **Startup type** of the Windows Installer service is **Manual**.

0x80200053

Expand table

Message	Description	Mitigation
BG_E_VALIDATION_FAILED	NA	Ensure that there are no firewalls that filter downloads. Such filtering could lead to incorrect responses being received by the Windows Update client.

0x80072EFD or 0x80072EFE or 0x80D02002

Expand table

Message	Description	Mitigation
TIME_OUT_ERRORS	The operation timed out	Make sure there are no firewall rules or proxies that block Microsoft download URLs. Take a network monitor trace to understand better. <Refer to Firewall Troubleshooting scenario>

0X8007000D

Expand table

Message	Description	Mitigation
ERROR_INVALID_DATA	Indicates data that isn't valid was downloaded or corruption occurred.	Attempt to redownload the update and start installation.

0x8024A10A

Expand table

Message	Description	Mitigation
USO_E_SERVICE_SHUTTING_DOWN	Indicates that the Windows Update Service is shutting down.	This error can occur after a long period of time of inactivity. The system fails to respond, leading to the service being idle and causing the service to shut down. Ensure that the system remains active and the connections remain established to complete the installation.

0x80240020

Expand table

Message	Description	Mitigation
WU_E_NO_INTERACTIVE_USER	Operation didn't complete because no interactive user is signed in.	Sign in to the device to start the installation and allow the device to restart.

0x80242014

Expand table

Message	Description	Mitigation
WU_E_UH_POSTREBOOTSTILLPENDING	The post-restart operation for the update is still in progress.	Some Windows updates require the device to be restarted. Restart the device to complete update installation.

0x80246017

Expand table

Message	Description	Mitigation
WU_E_DM_UNAUTHORIZED_LOCAL_USER	The download failed because the local user was denied authorization to download the content.	Ensure that the user attempting to download and install updates has been provided with sufficient privileges to install updates (Local Administrator).

0x8024000B

Expand table

Message	Description	Mitigation
WU_E_CALL_CANCELLED	Operation was canceled.	The operation was canceled by the user or service. You might also receive this error when we're unable to filter the results.

0x8024000E

Expand table

Message	Description	Mitigation
WU_E_XML_INVALID	Windows Update Agent found information in the update's XML data that isn't valid.	Certain drivers contain more metadata information in Update.xml, which Orchestrator can interpret as data that isn't valid. Ensure that you have the latest Windows Update Agent installed on the device.

0x8024D009

Expand table

Message	Description	Mitigation
WU_E_SETUP_SKIP_UPDATE	An update to the Windows Update Agent was skipped due to a directive in the Wuident.cab file.	You might encounter this error when WSUS isn't sending the self-update to the clients. For more information to resolve the issue, review KB920659.

0x80244007

Expand table

Message	Description	Mitigation
WU_E_PT_SOAPCLIENT_SOAPFAULT	SOAP client failed because there was a SOAP fault for reasons of `WU_E_PT_SOAP_*` error codes.	This issue occurs because Windows can't renew the cookies for Windows Update. For more information to resolve the issue, see 0x80244007 error when Windows tries to scan for updates on a WSUS server.

0x80070422

Expand table

Message	Description	Mitigation
ERROR_SERVICE_DISABLED	This issue occurs when the Windows Update service stops working or isn't running.	Check if the Windows Update service is running.

0x800f0821

Expand table

Message	Description	Mitigation
CBS_E_ABORT; client abort, IDABORT returned by ICbsUIHandler method except Error()	CBS transaction timeout exceeded.	A servicing operation is taking a long time to complete. The servicing stack watchdog timer expires. Extending the timeout will mitigate the issue. Increase the resources on the device. If a virtual machine, increase virtual CPU and memory to speed up operations. Make sure the device has installed the update in KB4493473 or later.

0x800f0825

Expand table

Message	Description	Mitigation

CBS_E_CANNOT_UNINSTALL; Package can't be uninstalled.	Typically this error is due component store corruption caused when a component is in a partially installed state.	Repair the component store with the `Dism RestoreHealth` command or manually repair with a payload from the partially installed component. From an elevated command prompt, run these commands:
		`Dism.exe /Online /Cleanup-Image /Restorehealth`
		`Sfc.exe /Scannow`
		Restart the device.

0x800F0920

Expand table

Message	Description	Mitigation
CBS_E_HANG_DETECTED; A failure to respond was detected while processing the operation.	Subsequent error logged after getting 0x800f0821	A servicing operation is taking a long time to complete. The servicing stack watchdog timer expires and assumes the system has stopped responding. Extending the timeout will mitigate the issue. Increase the resources on the device. If a virtual machine, increase virtual CPU and memory to speed up operations. Make sure the device has installed the update in KB4493473 or later.

0x800f081f

Expand table

Message	Description	Mitigation
CBS_E_SOURCE_MISSING; source for package or file not found, ResolveSource() unsuccessful	Component Store corruption	Repair the component store with the `Dism RestoreHealth` command or manually repair with the payload from the partially installed component. From an elevated command prompt and run these commands: `Dism.exe /Online /Cleanup-Image /Restorehealth`

Message	Description	Mitigation
		`Sfc.exe /Scannow` Restart the device.

0x800f0831

Expand table

Message	Description	Mitigation

CBS_E_STORE_CORRUPTION; CBS store is corrupted.	Corruption in the Windows Component Store.	Repair the component store with `Dism RestoreHealth` or manually repair with the payload from the partially installed component. From an elevated command prompt and run these commands:
		`Dism.exe /Online /Cleanup-Image /Restorehealth`
		`Sfc.exe /Scannow`
		Restart the device.

0x80070005

Expand table

Message	Description	Mitigation
E_ACCESSDENIED; General access denied error	File system or registry key permissions have been changed and the servicing stack doesn't have the required level of access.	This error generally means an access was denied. Go to *%Windir%\logs\CBS*, open the last *CBS.log* and search for `, error` and match with the timestamp. After finding the error, scroll up and try to determine what caused the access denial. It could be access denied to a file, registry key. Determine what object needs the right permissions and change the permissions as needed.

0x80070570

Expand table

Message	Description	Mitigation

ERROR_FILE_CORRUPT; The file or directory is corrupted and unreadable.	Component Store corruption	Repair the component store with `Dism RestoreHealth` or manually repair with the payload from the partially installed component. From an elevated command prompt and run these commands:
		`Dism.exe /Online /Cleanup-Image /Restorehealth`
		`Sfc.exe /Scannow`
		Restart the device.

0x80070003

Expand table

Message	Description	Mitigation
ERROR_PATH_NOT_FOUND; The system can't find the path specified.	The servicing stack can't access a specific path.	Indicates an invalid path to an executable. Go to *%Windir%\logs\CBS*, open the last *CBS.log*, and search for `, error`. Then match the results with the timestamp.

0x80070020

Expand table

Message	Description	Mitigation
ERROR_SHARING_VIOLATION	Numerous causes. CBS log analysis required.	This error is caused by non-Microsoft filter drivers like antivirus. 1. Perform a clean boot and retry the installation 2. Download the sysinternal tool Process Monitor. 3. Run *Procmon.exe*. It will start data capture automatically. 4. Install the update package again 5. With the Process Monitor main window in focus, press CTRL + E or select the magnifying glass to stop data capture. 6. Select **File** > **Save** > **All Events** > **PML**, and choose a path to save the `.PML` file 7. Go to *%windir%\logs\cbs*, open the last *Cbs.log* file, and search for the error. After finding the error line a bit above, you should have the file being accessed during the installation that is giving the sharing violation error 8. In Process Monitor, filter for path and insert the file name (it should be something like "path" "contains" "filename from CBS"). 9. Try to stop it or uninstall the process causing the error.

0x80073701

Expand table

Message	Description	Mitigation

ERROR_SXS_ASSEMBLY_MISSING; The referenced assembly couldn't be found.	Typically, a component store corruption caused when a component is in a partially installed state.	Repair the component store with `Dism RestoreHealth command` or manually repair it with the payload from the partially installed component. From an elevated command prompt, run these commands:
		`Dism.exe /Online /Cleanup-Image /Restorehealth`
		`Sfc.exe /Scannow`
		Restart the device.

0x8007371b

Expand table

Message	Description	Mitigation

ERROR_SXS_TRANSACTION_CLOSURE_INCOMPLETE; One or more required members of the transaction aren't present.	Component Store corruption.	Repair the component store with `Dism RestoreHealth` command or manually repair it with the payload from the partially installed component. From an elevated command prompt and run these commands:
		`Dism.exe /Online /Cleanup-Image /Restorehealth`
		`Sfc.exe /Scannow`
		Restart the device.

0x80072EFE

Expand table

Message	Description	Mitigation
WININET_E_CONNECTION_ABORTED; The connection with the server was closed abnormally	BITS is unable to transfer the file successfully.	Encountered if BITS is broken or if the file being transferred can't be written to the destination folder on the client. This error is caused by connection errors while checking or downloading updates. From a cmd prompt run: `BITSADMIN /LIST /ALLUSERS /VERBOSE` Search for the 0x80072EFE error code. You should see a reference to an HTTP code with a specific file. Using a browser, try to download it

Message	Description	Mitigation
		manually, making sure you're using your organization's proxy settings. If the download fails, check with your proxy manager to allow for the communication to be sucesfull. Also check with your network team for this specific URL access.

0x80072F8F

Expand table

Message	Description	Mitigation
WININET_E_DECODING_FAILED; Content decoding has failed	TLS 1.2 isn't configured correctly on the client.	This error generally means that the Windows Update Agent was unable to decode the received content. Install and configure TLS 1.2 by installing the update in KB3140245.

0x80072EE2

Expand table

Message	Description	Mitigation

WININET_E_TIMEOUT; The operation timed out	Unable to scan for updates due to a connectivity issue to Windows Update, Configuration Manager, or WSUS.	This error generally means that the Windows Update Agent was unable to connect to the update servers or your own source, such as WSUS, Configuration Manager, or Microsoft Intune.
		Check with your network team to ensure that the device can reach the update sources. For more info, see Troubleshoot software update scan failures in Configuration Manager.
		If you're using the public Microsoft update servers, check that your device can access the following Windows Update endpoints:
		`http://windowsupdate.microsoft.com`
		`https://*.windowsupdate.microsoft.com`
		`https://update.microsoft.com`
		`https://*.update.microsoft.com`
		`https://windowsupdate.com`
		`https://*.windowsupdate.com`
		`https://download.windowsupdate.com`
		`https://*.download.windowsupdate.com`
		`https://download.microsoft.com`
		`https://*.download.windowsupdate.com`
		`https://wustat.windows.com`
		`https://*.wustat.windows.com`
		`https://ntservicepack.microsoft.com`

0x80240022

Expand table

Message	Description	Mitigation
WU_E_ALL_UPDATES_FAILED; Operation failed for all the updates.	Multiple root causes for this error.	Most common issue is that antivirus software is blocking access to certain folders (like SoftwareDistribution). CBS.log analysis needed to determine the file or folder being protected.

0x8024401B

Expand table

Message	Description	Mitigation
WU_E_PT_HTTP_STATUS_PROXY_AUTH_REQ; Same as HTTP status 407 - proxy authentication is required.	Unable to authenticate through a proxy server.	Either the Winhttp proxy or WinInet proxy settings aren't configured correctly. This error generally means that the Windows Update Agent was unable to connect to the update servers or your own update source, such as WSUS, Configuration Manager, or Microsoft Intune, due to a proxy error. Verify the proxy settings on the client. The Windows Update Agent uses WinHTTP to scan for available updates. When there's a proxy server between the client and the update source, the proxy settings must be configured correctly on the clients to enable them to communicate by using the source's FQDN. Check with your network and proxy teams to confirm that the device can the update source without the proxy requiring user authentication.

0x80244022

Expand table

Message	Description	Mitigation
WU_E_PT_HTTP_STATUS_SERVICE_UNAVAILABLE; Same as HTTP status 503 - the service is temporarily overloaded.	Unable to connect to the configured update source.	Network troubleshooting needed to resolve the connectivity issue. Check with your network and proxy teams to confirm that the device can the update source without the proxy requiring user authentication.

0x80070490

Expand table

Message	Description	Mitigation
ERROR_NOT_FOUND	This error occurs during driver installation as part of the update.	This issue occurs because details such as the architecture for a driver that's being updated are missing in the registry. Manually add the missing inf file **Arch** value in the Driver operations registry by following these steps: 1. Open *regedit* and navigate to `HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\ComponentBasedServicing\DriverOperations\0\2(SequenceID)` 2. Review the **Identity** value to determine the value that is missing. 3. Manually add the missing value referring to the information in the **Identity** value. For example, Name: Arch; Type: REG_SZ (String Value); Data: amd64. 4. Proceed with installing the failing update.

0x800f0922

Expand table

Message	Description	Mitigation
CBS_E_INSTALLERS_FAILED	The July cumulative update failed to be installed on Windows Server 2016	In the CBS.log, you may find that updates sometimes roll back when License and Product key tokens fail to be updated. This issue can be resolved by adding write permissions for the "User" and "Network Service" accounts to the *C:\Windows\System32\spp* folder.

0x80070bc9

Expand table

Message	Description	Mitigation
ERROR_FAIL_REBOOT_REQUIRED	The TrustedInstaller service startup type is set to "Manual" by Group Policy (GPO), which prevented it from starting to complete pending operations.	The TrustedInstaller service changes the startup type from Manual to Automatic when it encounters an update that has to process a transaction after a restart. When the value is set back to Manual before the restart, the transaction cannot be applied. This transaction will be pending and block all other update installations. To fix this issue, change the TrustedInstaller policy to **Automatic** and restart the computer. If it doesn't work, start the computer to WinRE to revert the pending actions. For example, `dism /Image:C:\ /Cleanup-Image /RevertPendingActions`. If it doesn't work either, start the computer to WinRE, rename *\WinSxS\Pending.xml*, and remove the PendingXMLIdentifier from COMPONENTS Hive.

0x800706be

Expand table

Message	Description	Mitigation
Failed to install cumulative updates	Windows Server 2016 Std failed to install cumulative packages by using the .msu package. No error is returned. When installing the packages with dism.exe, it returned the error 0x800706be.	The last cumulative update failed to install and was corrupted. To resolve this issue, navigate to the registry key for the corrupted update package. Change the "current state" value to 000020 hex (32 dec) - resolved, or 000040 hex (64 dec) - staged, or 000070 hex (112 dec) - installed.

Data collection

If you need assistance from Microsoft support, we recommend you collect the information by following the steps mentioned in Gather information by using TSS for deployment-related issues.

Windows Update error codes by component

Article • 08/29/2023 • Applies to: ✅ Windows 11, ✅ Windows 10

This section lists the error codes for Microsoft Windows Update.

Automatic Update Errors

Expand table

Error code	Message	Description
`0x80243FFF`	`WU_E_AUCLIENT_UNEXPECTED`	There was a user interface error not covered by another `WU_E_AUCLIENT_*` error code.
`0x8024A000`	`WU_E_AU_NOSERVICE`	Automatic Updates was unable to service incoming requests.
`0x8024A002`	`WU_E_AU_NONLEGACYSERVER`	The old version of the Automatic Updates client has stopped because the WSUS server has been upgraded.
`0x8024A003`	`WU_E_AU_LEGACYCLIENTDISABLED`	The old version of the Automatic Updates client was disabled.
`0x8024A004`	`WU_E_AU_PAUSED`	Automatic Updates was unable to process incoming requests because it was paused.
`0x8024A005`	`WU_E_AU_NO_REGISTERED_SERVICE`	No unmanaged service is registered with `AU`.
`0x8024AFFF`	`WU_E_AU_UNEXPECTED`	An Automatic Updates error not covered by another `WU_E_AU*` code.

Windows Update UI errors

Expand table

Error code	Message	Description
`0x80243001`	`WU_E_INSTALLATION_RESULTS_UNKNOWN_VERSION`	The results of download and installation couldn't be read from the registry due to an unrecognized data format version.

Error code	Message	Description
`0x80243002`	`WU_E_INSTALLATION_RESULTS_INVALID_DATA`	The results of download and installation couldn't be read from the registry due to an invalid data format.
`0x80243003`	`WU_E_INSTALLATION_RESULTS_NOT_FOUND`	The results of download and installation aren't available; the operation may have failed to start.
`0x80243004`	`WU_E_TRAYICON_FAILURE`	A failure occurred when trying to create an icon in the taskbar notification area.
`0x80243FFD`	`WU_E_NON_UI_MODE`	Unable to show UI when in non-UI mode; Windows Update client UI modules may not be installed.
`0x80243FFE`	`WU_E_WUCLTUI_UNSUPPORTED_VERSION`	Unsupported version of Windows Update client UI exported functions.
`0x80243FFF`	`WU_E_AUCLIENT_UNEXPECTED`	There was a user interface error not covered by another `WU_E_AUCLIENT_*` error code.
`0x8024043D`	`WU_E_SERVICEPROP_NOTAVAIL`	The requested service property isn't available.

Inventory errors

Expand table

Error code	Message	Description
`0x80249001`	`WU_E_INVENTORY_PARSEFAILED`	Parsing of the rule file failed.
`0x80249002`	`WU_E_INVENTORY_GET_INVENTORY_TYPE_FAILED`	Failed to get the requested inventory type from the server.
`0x80249003`	`WU_E_INVENTORY_RESULT_UPLOAD_FAILED`	Failed to upload inventory result to the server.
`0x80249004`	`WU_E_INVENTORY_UNEXPECTED`	There was an inventory error not covered by another error code.
`0x80249005`	`WU_E_INVENTORY_WMI_ERROR`	A WMI error occurred when enumerating the instances for a particular class.

Expression evaluator errors

Expand table

Error code	Message	Description
`0x8024E001`	`WU_E_EE_UNKNOWN_EXPRESSION`	An expression evaluator operation couldn't be completed because an expression was unrecognized.
`0x8024E002`	`WU_E_EE_INVALID_EXPRESSION`	An expression evaluator operation couldn't be completed because an expression was invalid.
`0x8024E003`	`WU_E_EE_MISSING_METADATA`	An expression evaluator operation couldn't be completed because an expression contains an incorrect number of metadata nodes.
`0x8024E004`	`WU_E_EE_INVALID_VERSION`	An expression evaluator operation couldn't be completed because the version of the serialized expression data is invalid.
`0x8024E005`	`WU_E_EE_NOT_INITIALIZED`	The expression evaluator couldn't be initialized.
`0x8024E006`	`WU_E_EE_INVALID_ATTRIBUTEDATA`	An expression evaluator operation couldn't be completed because there was an invalid attribute.
`0x8024E007`	`WU_E_EE_CLUSTER_ERROR`	An expression evaluator operation couldn't be completed because the cluster state of the computer couldn't be determined.
`0x8024EFFF`	`WU_E_EE_UNEXPECTED`	There was an expression evaluator error not covered by another `WU_E_EE_*` error code.

Reporter errors

Expand table

Error code	Message	Description
`0x80247001`	`WU_E_OL_INVALID_SCANFILE`	An operation couldn't be completed because the scan package was invalid.
`0x80247002`	`WU_E_OL_NEWCLIENT_REQUIRED`	An operation couldn't be completed because the scan package requires a greater version of the Windows Update Agent.
`0x80247FFF`	`WU_E_OL_UNEXPECTED`	Search using the scan package failed.

Error code	Message	Description
`0x8024F001`	`WU_E_REPORTER_EVENTCACHECORRUPT`	The event cache file was defective.
`0x8024F002`	`WU_E_REPORTER_EVENTNAMESPACEPARSEFAILED`	The XML in the event namespace descriptor couldn't be parsed.
`0x8024F003`	`WU_E_INVALID_EVENT`	The XML in the event namespace descriptor couldn't be parsed.
`0x8024F004`	`WU_E_SERVER_BUSY`	The server rejected an event because the server was too busy.
`0x8024FFFF`	`WU_E_REPORTER_UNEXPECTED`	There was a reporter error not covered by another error code.

Redirector errors

The components that download the `Wuredir.cab` file and then parse the `Wuredir.cab` file generate the following errors.

Expand table

Error code	Message	Description
`0x80245001`	`WU_E_REDIRECTOR_LOAD_XML`	The redirector XML document couldn't be loaded into the DOM class.
`0x80245002`	`WU_E_REDIRECTOR_S_FALSE`	The redirector XML document is missing some required information.
`0x80245003`	`WU_E_REDIRECTOR_ID_SMALLER`	The redirectorId in the downloaded redirector cab is less than in the cached cab.
`0x80245FFF`	`WU_E_REDIRECTOR_UNEXPECTED`	The redirector failed for reasons not covered by another `WU_E_REDIRECTOR_*` error code.

Protocol Talker errors

The following errors map to `SOAPCLIENT_ERROR`s through the `Atlsoap.h` file. These errors are obtained when the `CClientWebService` object calls the `GetClientError()` method.

Expand table

Error code	Message	Description
`0x80244000`	`WU_E_PT_SOAPCLIENT_BASE`	`WU_E_PT_SOAPCLIENT_*` error codes map to the `SOAPCLIENT_ERROR` enum of the ATL Server Library.
`0x80244001`	`WU_E_PT_SOAPCLIENT_INITIALIZE`	Same as `SOAPCLIENT_INITIALIZE_ERROR` - initialization of the `SOAP` client failed possibly because of an MSXML installation failure.
`0x80244002`	`WU_E_PT_SOAPCLIENT_OUTOFMEMORY`	Same as `SOAPCLIENT_OUTOFMEMORY` - `SOAP` client failed because it ran out of memory.
`0x80244003`	`WU_E_PT_SOAPCLIENT_GENERATE`	Same as `SOAPCLIENT_GENERATE_ERROR` - `SOAP` client failed to generate the request.
`0x80244004`	`WU_E_PT_SOAPCLIENT_CONNECT`	Same as `SOAPCLIENT_CONNECT_ERROR` - `SOAP` client failed to connect to the server.
`0x80244005`	`WU_E_PT_SOAPCLIENT_SEND`	Same as `SOAPCLIENT_SEND_ERROR` - `SOAP` client failed to send a message for reasons of `WU_E_WINHTTP_*` error codes.
`0x80244006`	`WU_E_PT_SOAPCLIENT_SERVER`	Same as `SOAPCLIENT_SERVER_ERROR` - `SOAP` client failed because there was a server error.
`0x80244007`	`WU_E_PT_SOAPCLIENT_SOAPFAULT`	Same as `SOAPCLIENT_SOAPFAULT` - `SOAP` client failed because there was a SOAP fault for reasons of `WU_E_PT_SOAP_*` error codes.
`0x80244008`	`WU_E_PT_SOAPCLIENT_PARSEFAULT`	Same as `SOAPCLIENT_PARSEFAULT_ERROR` - `SOAP` client failed to parse a `SOAP` fault.
`0x80244009`	`WU_E_PT_SOAPCLIENT_READ`	Same as `SOAPCLIENT_READ_ERROR` - `SOAP` client failed while reading the response from the server.
`x8024400A`	`WU_E_PT_SOAPCLIENT_PARSE`	Same as `SOAPCLIENT_PARSE_ERROR` - `SOAP` client failed to parse the response from the server.

Other Protocol Talker errors

The following errors map to `SOAP_ERROR_CODE`s from the `Atlsoap.h` file. These errors are obtained from the `m_fault.m_soapErrCode` member of the `CClientWebService` object when `GetClientError()` returns `SOAPCLIENT_SOAPFAULT`.

Expand table

Error code	Message	Description
`0x8024400B`	`WU_E_PT_SOAP_VERSION`	Same as `SOAP_E_VERSION_MISMATCH` - `SOAP` client found an unrecognizable namespace for the `SOAP` envelope.
`0x8024400C`	`WU_E_PT_SOAP_MUST_UNDERSTAND`	Same as `SOAP_E_MUST_UNDERSTAND` - `SOAP` client was unable to understand a header.
`0x8024400D`	`WU_E_PT_SOAP_CLIENT`	Same as `SOAP_E_CLIENT` - `SOAP` client found the message was malformed; fix before resending.
`0x8024400E`	`WU_E_PT_SOAP_SERVER`	Same as `SOAP_E_SERVER` - The `SOAP` message couldn't be processed due to a server error; resend later.
`0x8024400F`	`WU_E_PT_WMI_ERROR`	There was an unspecified Windows Management Instrumentation (WMI) error.
`0x80244010`	`WU_E_PT_EXCEEDED_MAX_SERVER_TRIPS`	The number of round trips to the server exceeded the maximum limit.
`0x80244011`	`WU_E_PT_SUS_SERVER_NOT_SET`	WUServer policy value is missing in the registry.
`0x80244012`	`WU_E_PT_DOUBLE_INITIALIZATION`	Initialization failed because the object was already initialized.
`0x80244013`	`WU_E_PT_INVALID_COMPUTER_NAME`	The computer name couldn't be determined.
`0x80244015`	`WU_E_PT_REFRESH_CACHE_REQUIRED`	The reply from the server indicates that the server was changed or the cookie was invalid; refresh the state of the internal cache and retry.
`0x80244016`	`WU_E_PT_HTTP_STATUS_BAD_REQUEST`	Same as HTTP status 400 - the server couldn't process the request due to invalid syntax.
`0x80244017`	`WU_E_PT_HTTP_STATUS_DENIED`	Same as HTTP status 401 - the requested resource requires user authentication.
`0x80244018`	`WU_E_PT_HTTP_STATUS_FORBIDDEN`	Same as HTTP status 403 - server understood the request but declined to fulfill it.
`0x80244019`	`WU_E_PT_HTTP_STATUS_NOT_FOUND`	Same as HTTP status 404 - the server can't find the requested URI (Uniform

Error code	Message	Description
		Resource Identifier).
`0x8024401A`	`WU_E_PT_HTTP_STATUS_BAD_METHOD`	Same as HTTP status 405 - the HTTP method isn't allowed.
`0x8024401B`	`WU_E_PT_HTTP_STATUS_PROXY_AUTH_REQ`	Same as HTTP status 407 - proxy authentication is required.
`0x8024401C`	`WU_E_PT_HTTP_STATUS_REQUEST_TIMEOUT`	Same as HTTP status 408 - the server timed out waiting for the request.
`0x8024401D`	`WU_E_PT_HTTP_STATUS_CONFLICT`	Same as HTTP status 409 - the request wasn't completed due to a conflict with the current state of the resource.
`0x8024401E`	`WU_E_PT_HTTP_STATUS_GONE`	Same as HTTP status 410 - requested resource is no longer available at the server.
`0x8024401F`	`WU_E_PT_HTTP_STATUS_SERVER_ERROR`	Same as HTTP status 500 - an error internal to the server prevented fulfilling the request.
`0x80244020`	`WU_E_PT_HTTP_STATUS_NOT_SUPPORTED`	Same as HTTP status 500 - server doesn't support the functionality required to fulfill the request.
`0x80244021`	`WU_E_PT_HTTP_STATUS_BAD_GATEWAY`	Same as HTTP status 502 - the server while acting as a gateway or a proxy received an invalid response from the upstream server it accessed in attempting to fulfill the request.
`0x80244022`	`WU_E_PT_HTTP_STATUS_SERVICE_UNAVAIL`	Same as HTTP status 503 - the service is temporarily overloaded.
`0x80244023`	`WU_E_PT_HTTP_STATUS_GATEWAY_TIMEOUT`	Same as HTTP status 503 - the request was timed out waiting for a gateway.
`0x80244024`	`WU_E_PT_HTTP_STATUS_VERSION_NOT_SUP`	Same as HTTP status 505 - the server doesn't support the HTTP protocol version used for the request.
`0x80244025`	`WU_E_PT_FILE_LOCATIONS_CHANGED`	Operation failed due to a changed file location; refresh internal state and resend.
`0x80244026`	`WU_E_PT_REGISTRATION_NOT_SUPPORTED`	Operation failed because Windows Update Agent doesn't support registration with a non-WSUS server.

Error code	Message	Description
`0x80244027`	`WU_E_PT_NO_AUTH_PLUGINS_REQUESTED`	The server returned an empty authentication information list.
`0x80244028`	`WU_E_PT_NO_AUTH_COOKIES_CREATED`	Windows Update Agent was unable to create any valid authentication cookies.
`0x80244029`	`WU_E_PT_INVALID_CONFIG_PROP`	A configuration property value was wrong.
`0x8024402A`	`WU_E_PT_CONFIG_PROP_MISSING`	A configuration property value was missing.
`0x8024402B`	`WU_E_PT_HTTP_STATUS_NOT_MAPPED`	The HTTP request couldn't be completed and the reason didn't correspond to any of the `WU_E_PT_HTTP_*` error codes.
`0x8024402C`	`WU_E_PT_WINHTTP_NAME_NOT_RESOLVED`	Same as ERROR_WINHTTP_NAME_NOT_RESOLVED - the proxy server or target server name can't be resolved.
`0x8024402F`	`WU_E_PT_ECP_SUCCEEDED_WITH_ERRORS`	External cab file processing completed with some errors.
`0x80244030`	`WU_E_PT_ECP_INIT_FAILED`	The external cab processor initialization didn't complete.
`0x80244031`	`WU_E_PT_ECP_INVALID_FILE_FORMAT`	The format of a metadata file was invalid.
`0x80244032`	`WU_E_PT_ECP_INVALID_METADATA`	External cab processor found invalid metadata.
`0x80244033`	`WU_E_PT_ECP_FAILURE_TO_EXTRACT_DIGEST`	The file digest couldn't be extracted from an external cab file.
`0x80244034`	`WU_E_PT_ECP_FAILURE_TO_DECOMPRESS_CAB_FILE`	An external cab file couldn't be decompressed.
`0x80244035`	`WU_E_PT_ECP_FILE_LOCATION_ERROR`	External cab processor was unable to get file locations.
`0x80244FFF`	`WU_E_PT_UNEXPECTED`	A communication error not covered by another `WU_E_PT_*` error code.
`0x8024502D`	`WU_E_PT_SAME_REDIR_ID`	Windows Update Agent failed to download a redirector cabinet file with a new redirectorId value from the server during the recovery.

Error code	Message	Description
`0x8024502E`	`WU_E_PT_NO_MANAGED_RECOVER`	A redirector recovery action didn't complete because the server is managed.

Download Manager errors

Expand table

Error code	Message	Description
`0x80246001`	`WU_E_DM_URLNOTAVAILABLE`	A download manager operation couldn't be completed because the requested file doesn't have a URL.
`0x80246002`	`WU_E_DM_INCORRECTFILEHASH`	A download manager operation couldn't be completed because the file digest wasn't recognized.
`0x80246003`	`WU_E_DM_UNKNOWNALGORITHM`	A download manager operation couldn't be completed because the file metadata requested an unrecognized hash algorithm.
`0x80246004`	`WU_E_DM_NEEDDOWNLOADREQUEST`	An operation couldn't be completed because a download request is required from the download handler.
`0x80246005`	`WU_E_DM_NONETWORK`	A download manager operation couldn't be completed because the network connection was unavailable.
`0x80246006`	`WU_E_DM_WRONGBITSVERSION`	A download manager operation couldn't be completed because the version of Background Intelligent Transfer Service (BITS) is incompatible.
`0x80246007`	`WU_E_DM_NOTDOWNLOADED`	The update hasn't been downloaded.
`0x80246008`	`WU_E_DM_FAILTOCONNECTTOBITS`	A download manager operation failed because the download manager was unable to connect the Background Intelligent Transfer Service (BITS).
`0x80246009`	`WU_E_DM_BITSTRANSFERERROR`	A download manager operation failed because there was an unspecified Background Intelligent Transfer Service (BITS) transfer error.
`0x8024600A`	`WU_E_DM_DOWNLOADLOCATIONCHANGED`	A download must be restarted because the location of the source of the download has

Error code	Message	Description
		changed.
`0x8024600B`	`WU_E_DM_CONTENTCHANGED`	A download must be restarted because the update content changed in a new revision.
`0x80246FFF`	`WU_E_DM_UNEXPECTED`	There was a download manager error not covered by another `WU_E_DM_*` error code.

Update Handler errors

Expand table

Error code	Message	Description
`0x80242000`	`WU_E_UH_REMOTEUNAVAILABLE`	A request for a remote update handler couldn't be completed because no remote process is available.
`0x80242001`	`WU_E_UH_LOCALONLY`	A request for a remote update handler couldn't be completed because the handler is local only.
`0x80242002`	`WU_E_UH_UNKNOWNHANDLER`	A request for an update handler couldn't be completed because the handler couldn't be recognized.
`0x80242003`	`WU_E_UH_REMOTEALREADYACTIVE`	A remote update handler couldn't be created because one already exists.
`0x80242004`	`WU_E_UH_DOESNOTSUPPORTACTION`	A request for the handler to install (uninstall) an update couldn't be completed because the update doesn't support install (uninstall).
`0x80242005`	`WU_E_UH_WRONGHANDLER`	An operation didn't complete because the wrong handler was specified.
`0x80242006`	`WU_E_UH_INVALIDMETADATA`	A handler operation couldn't be completed because the update contains invalid metadata.
`0x80242007`	`WU_E_UH_INSTALLERHUNG`	An operation couldn't be completed because the installer exceeded the time limit.
`0x80242008`	`WU_E_UH_OPERATIONCANCELLED`	An operation being done by the update handler was canceled.

Error code	Message	Description
`0x80242009`	`WU_E_UH_BADHANDLERXML`	An operation couldn't be completed because the handler-specific metadata is invalid.
`0x8024200A`	`WU_E_UH_CANREQUIREINPUT`	A request to the handler to install an update couldn't be completed because the update requires user input.
`0x8024200B`	`WU_E_UH_INSTALLERFAILURE`	The installer failed to install (uninstall) one or more updates.
`0x8024200C`	`WU_E_UH_FALLBACKTOSELFCONTAINED`	The update handler should download self-contained content rather than delta-compressed content for the update.
`0x8024200D`	`WU_E_UH_NEEDANOTHERDOWNLOAD`	The update handler didn't install the update because it needs to be downloaded again.
`0x8024200E`	`WU_E_UH_NOTIFYFAILURE`	The update handler failed to send notification of the status of the install (uninstall) operation.
`0x8024200F`	`WU_E_UH_INCONSISTENT_FILE_NAMES`	The file names contained in the update metadata and in the update package are inconsistent.
`0x80242010`	`WU_E_UH_FALLBACKERROR`	The update handler failed to fall back to the self-contained content.
`0x80242011`	`WU_E_UH_TOOMANYDOWNLOADREQUESTS`	The update handler has exceeded the maximum number of download requests.
`0x80242012`	`WU_E_UH_UNEXPECTEDCBSRESPONSE`	The update handler has received an unexpected response from CBS.
`0x80242013`	`WU_E_UH_BADCBSPACKAGEID`	The update metadata contains an invalid CBS package identifier.
`0x80242014`	`WU_E_UH_POSTREBOOTSTILLPENDING`	The post-reboot operation for the update is still in progress.
`0x80242015`	`WU_E_UH_POSTREBOOTRESULTUNKNOWN`	The result of the post-reboot operation for the update couldn't be determined.
`0x80242016`	`WU_E_UH_POSTREBOOTUNEXPECTEDSTATE`	The state of the update after its post-reboot operation has completed is unexpected.

Error code	Message	Description
`0x80242017`	`WU_E_UH_NEW_SERVICING_STACK_REQUIRED`	The OS servicing stack must be updated before this update is downloaded or installed.
`0x80242FFF`	`WU_E_UH_UNEXPECTED`	An update handler error not covered by another `WU_E_UH_*` code.

Data Store errors

Expand table

Error code	Message	Description
`0x80248000`	`WU_E_DS_SHUTDOWN`	An operation failed because Windows Update Agent is shutting down.
`0x80248001`	`WU_E_DS_INUSE`	An operation failed because the data store was in use.
`0x80248002`	`WU_E_DS_INVALID`	The current and expected states of the data store don't match.
`0x80248003`	`WU_E_DS_TABLEMISSING`	The data store is missing a table.
`0x80248004`	`WU_E_DS_TABLEINCORRECT`	The data store contains a table with unexpected columns.
`0x80248005`	`WU_E_DS_INVALIDTABLENAME`	A table couldn't be opened because the table isn't in the data store.
`0x80248006`	`WU_E_DS_BADVERSION`	The current and expected versions of the data store don't match.
`0x80248007`	`WU_E_DS_NODATA`	The information requested isn't in the data store.
`0x80248008`	`WU_E_DS_MISSINGDATA`	The data store is missing required information or has a NULL in a table column that requires a non-null value.
`0x80248009`	`WU_E_DS_MISSINGREF`	The data store is missing required information or has a reference to missing license terms file localized property or linked row.
`0x8024800A`	`WU_E_DS_UNKNOWNHANDLER`	The update wasn't processed because its update handler couldn't be recognized.

Error code	Message	Description
`0x8024800B`	`WU_E_DS_CANTDELETE`	The update wasn't deleted because it's still referenced by one or more services.
`0x8024800C`	`WU_E_DS_LOCKTIMEOUTEXPIRED`	The data store section couldn't be locked within the allotted time.
`0x8024800D`	`WU_E_DS_NOCATEGORIES`	The category wasn't added because it contains no parent categories and isn't a top-level category itself.
`0x8024800E`	`WU_E_DS_ROWEXISTS`	The row wasn't added because an existing row has the same primary key.
`0x8024800F`	`WU_E_DS_STOREFILELOCKED`	The data store couldn't be initialized because it was locked by another process.
`0x80248010`	`WU_E_DS_CANNOTREGISTER`	The data store isn't allowed to be registered with COM in the current process.
`0x80248011`	`WU_E_DS_UNABLETOSTART`	Couldn't create a data store object in another process.
`0x80248013`	`WU_E_DS_DUPLICATEUPDATEID`	The server sent the same update to the client with two different revision IDs.
`0x80248014`	`WU_E_DS_UNKNOWNSERVICE`	An operation didn't complete because the service isn't in the data store.
`0x80248015`	`WU_E_DS_SERVICEEXPIRED`	An operation didn't complete because the registration of the service has expired.
`0x80248016`	`WU_E_DS_DECLINENOTALLOWED`	A request to hide an update was declined because it's a mandatory update or because it was deployed with a deadline.
`0x80248017`	`WU_E_DS_TABLESESSIONMISMATCH`	A table wasn't closed because it isn't associated with the session.
`0x80248018`	`WU_E_DS_SESSIONLOCKMISMATCH`	A table wasn't closed because it isn't associated with the session.
`0x80248019`	`WU_E_DS_NEEDWINDOWSSERVICE`	A request to remove the Windows Update service or to unregister it with Automatic Updates was declined because it's a built-in service and/or Automatic Updates can't fall back to another service.
`0x8024801A`	`WU_E_DS_INVALIDOPERATION`	A request was declined because the operation isn't allowed.

Error code	Message	Description
`0x8024801B`	`WU_E_DS_SCHEMAMISMATCH`	The schema of the current data store and the schema of a table in a backup XML document don't match.
`0x8024801C`	`WU_E_DS_RESETREQUIRED`	The data store requires a session reset; release the session and retry with a new session.
`0x8024801D`	`WU_E_DS_IMPERSONATED`	A data store operation didn't complete because it was requested with an impersonated identity.
`0x80248FFF`	`WU_E_DS_UNEXPECTED`	A data store error not covered by another `WU_E_DS_*` code.

Driver Util errors

The PnP enumerated device is removed from the System Spec because one of the hardware IDs or the compatible IDs matches an installed printer driver. This isn't a fatal error, and the device is merely skipped.

Expand table

Error code	Message	Description
`0x8024C001`	`WU_E_DRV_PRUNED`	A driver was skipped.
`0x8024C002`	`WU_E_DRV_NOPROP_OR_LEGACY`	A property for the driver couldn't be found. It may not conform with required specifications.
`0x8024C003`	`WU_E_DRV_REG_MISMATCH`	The registry type read for the driver doesn't match the expected type.
`0x8024C004`	`WU_E_DRV_NO_METADATA`	The driver update is missing metadata.
`0x8024C005`	`WU_E_DRV_MISSING_ATTRIBUTE`	The driver update is missing a required attribute.
`0x8024C006`	`WU_E_DRV_SYNC_FAILED`	Driver synchronization failed.
`0x8024C007`	`WU_E_DRV_NO_PRINTER_CONTENT`	Information required for the synchronization of applicable printers is missing.
`0x8024CFFF`	`WU_E_DRV_UNEXPECTED`	A driver error not covered by another `WU_E_DRV_*` code.

Windows Update error codes

Expand table

Error code	Message	Description
`0x80070020`	`InstallFileLocked`	Couldn't access the file because it is already in use. This can occur when the installer tries to replace a file that an antivirus, antimalware or backup program is currently scanning.
`0x80240001`	`WU_E_NO_SERVICE`	Windows Update Agent was unable to provide the service.
`0x80240002`	`WU_E_MAX_CAPACITY_REACHED`	The maximum capacity of the service was exceeded.
`0x80240003`	`WU_E_UNKNOWN_ID`	An ID can't be found.
`0x80240004`	`WU_E_NOT_INITIALIZED`	The object couldn't be initialized.
`0x80240005`	`WU_E_RANGEOVERLAP`	The update handler requested a byte range overlapping a previously requested range.
`0x80240006`	`WU_E_TOOMANYRANGES`	The requested number of byte ranges exceeds the maximum number (2^31 - 1).
`0x80240007`	`WU_E_INVALIDINDEX`	The index to a collection was invalid.
`0x80240008`	`WU_E_ITEMNOTFOUND`	The key for the item queried couldn't be found.
`0x80240009`	`WU_E_OPERATIONINPROGRESS`	Another conflicting operation was in progress. Some operations such as installation can't be performed twice simultaneously.
`0x8024000A`	`WU_E_COULDNOTCANCEL`	Cancellation of the operation wasn't allowed.
`0x8024000B`	`WU_E_CALL_CANCELLED`	Operation was canceled.
`0x8024000C`	`WU_E_NOOP`	No operation was required.
`0x8024000D`	`WU_E_XML_MISSINGDATA`	Windows Update Agent couldn't find required information in the update's XML data.
`0x8024000E`	`WU_E_XML_INVALID`	Windows Update Agent found invalid information in the update's XML data.
`0x8024000F`	`WU_E_CYCLE_DETECTED`	Circular update relationships were detected in the metadata.
`0x80240010`	`WU_E_TOO_DEEP_RELATION`	Update relationships too deep to evaluate were evaluated.
`0x80240011`	`WU_E_INVALID_RELATIONSHIP`	An invalid update relationship was detected.

Error code	Message	Description
`0x80240012`	`WU_E_REG_VALUE_INVALID`	An invalid registry value was read.
`0x80240013`	`WU_E_DUPLICATE_ITEM`	Operation tried to add a duplicate item to a list.
`0x80240016`	`WU_E_INSTALL_NOT_ALLOWED`	Operation tried to install while another installation was in progress or the system was pending a mandatory restart.
`0x80240017`	`WU_E_NOT_APPLICABLE`	Operation wasn't performed because there are no applicable updates.
`0x80240018`	`WU_E_NO_USERTOKEN`	Operation failed because a required user token is missing.
`0x80240019`	`WU_E_EXCLUSIVE_INSTALL_CONFLICT`	An exclusive update can't be installed with other updates at the same time.
`0x8024001A`	`WU_E_POLICY_NOT_SET`	A policy value wasn't set.
`0x8024001B`	`WU_E_SELFUPDATE_IN_PROGRESS`	The operation couldn't be performed because the Windows Update Agent is self-updating.
`0x8024001D`	`WU_E_INVALID_UPDATE`	An update contains invalid metadata.
`0x8024001E`	`WU_E_SERVICE_STOP`	Operation didn't complete because the service or system was being shut down.
`0x8024001F`	`WU_E_NO_CONNECTION`	Operation didn't complete because the network connection was unavailable.
`0x80240020`	`WU_E_NO_INTERACTIVE_USER`	Operation didn't complete because there's no logged-on interactive user.
`0x80240021`	`WU_E_TIME_OUT`	Operation didn't complete because it timed out.
`0x80240022`	`WU_E_ALL_UPDATES_FAILED`	Operation failed for all the updates.
`0x80240023`	`WU_E_EULAS_DECLINED`	The license terms for all updates were declined.
`0x80240024`	`WU_E_NO_UPDATE`	There are no updates.
`0x80240025`	`WU_E_USER_ACCESS_DISABLED`	Group Policy settings prevented access to Windows Update.
`0x80240026`	`WU_E_INVALID_UPDATE_TYPE`	The type of update is invalid.
`0x80240027`	`WU_E_URL_TOO_LONG`	The URL exceeded the maximum length.
`0x80240028`	`WU_E_UNINSTALL_NOT_ALLOWED`	The update couldn't be uninstalled because the

Error code	Message	Description
		request didn't originate from a WSUS server.
`0x80240029`	`WU_E_INVALID_PRODUCT_LICENSE`	Search may have missed some updates before there's an unlicensed application on the system.
`0x8024002A`	`WU_E_MISSING_HANDLER`	A component required to detect applicable updates was missing.
`0x8024002B`	`WU_E_LEGACYSERVER`	An operation didn't complete because it requires a newer version of server.
`0x8024002C`	`WU_E_BIN_SOURCE_ABSENT`	A delta-compressed update couldn't be installed because it required the source.
`0x8024002D`	`WU_E_SOURCE_ABSENT`	A full-file update couldn't be installed because it required the source.
`0x8024002E`	`WU_E_WU_DISABLED`	Access to an unmanaged server isn't allowed.
`0x8024002F`	`WU_E_CALL_CANCELLED_BY_POLICY`	Operation didn't complete because the DisableWindowsUpdateAccess policy was set.
`0x80240030`	`WU_E_INVALID_PROXY_SERVER`	The format of the proxy list was invalid.
`0x80240031`	`WU_E_INVALID_FILE`	The file is in the wrong format.
`0x80240032`	`WU_E_INVALID_CRITERIA`	The search criteria string was invalid.
`0x80240033`	`WU_E_EULA_UNAVAILABLE`	License terms couldn't be downloaded.
`0x80240034`	`WU_E_DOWNLOAD_FAILED`	Update failed to download.
`0x80240035`	`WU_E_UPDATE_NOT_PROCESSED`	The update wasn't processed.
`0x80240036`	`WU_E_INVALID_OPERATION`	The object's current state didn't allow the operation.
`0x80240037`	`WU_E_NOT_SUPPORTED`	The functionality for the operation isn't supported.
`0x80240038`	`WU_E_WINHTTP_INVALID_FILE`	The downloaded file has an unexpected content type.
`0x80240039`	`WU_E_TOO_MANY_RESYNC`	Agent is asked by server to resync too many times.
`0x80240040`	`WU_E_NO_SERVER_CORE_SUPPORT`	`WUA API` method doesn't run on Server Core installation.
`0x80240041`	`WU_E_SYSPREP_IN_PROGRESS`	Service isn't available while sysprep is running.

Error code	Message	Description
`0x80240042`	`WU_E_UNKNOWN_SERVICE`	The update service is no longer registered with `AU`.
`0x80240043`	`WU_E_NO_UI_SUPPORT`	There's no support for `WUA UI`.
`0x80240FFF`	`WU_E_UNEXPECTED`	An operation failed due to reasons not covered by another error code.
`0x80070422`		Windows Update service stopped working or isn't running.

Windows Update success codes

Expand table

Error code	Message	Description
`0x00240001`	`WU_S_SERVICE_STOP`	Windows Update Agent was stopped successfully.
`0x00240002`	`WU_S_SELFUPDATE`	Windows Update Agent updated itself.
`0x00240003`	`WU_S_UPDATE_ERROR`	Operation completed successfully but there were errors applying the updates.
`0x00240004`	`WU_S_MARKED_FOR_DISCONNECT`	A callback was marked to be disconnected later because the request to disconnect the operation came while a callback was executing.
`0x00240005`	`WU_S_REBOOT_REQUIRED`	The system must be restarted to complete installation of the update.
`0x00240006`	`WU_S_ALREADY_INSTALLED`	The update to be installed is already installed on the system.
`0x00240007`	`WU_S_ALREADY_UNINSTALLED`	The update to be removed isn't installed on the system.
`0x00240008`	`WU_S_ALREADY_DOWNLOADED`	The update to be downloaded has already been downloaded.

Windows Installer minor errors

The following errors are used to indicate that part of a search fails because of Windows Installer problems. Another part of the search may successfully return updates. All

Windows Installer minor codes must share the same error code range so that the caller can tell that they're related to Windows Installer.

Expand table

Error code	Message	Description
`0x80241001`	`WU_E_MSI_WRONG_VERSION`	Search may have missed some updates because the Windows Installer is less than version 3.1.
`0x80241002`	`WU_E_MSI_NOT_CONFIGURED`	Search may have missed some updates because the Windows Installer isn't configured.
`0x80241003`	`WU_E_MSP_DISABLED`	Search may have missed some updates because policy has disabled Windows Installer patching.
`0x80241004`	`WU_E_MSI_WRONG_APP_CONTEXT`	An update couldn't be applied because the application is installed per-user.
`0x80241FFF`	`WU_E_MSP_UNEXPECTED`	Search may have missed some updates because there was a failure of the Windows Installer.

Windows Update Agent update and setup errors

Expand table

Error code	Message	Description
`0x8024D001`	`WU_E_SETUP_INVALID_INFDATA`	Windows Update Agent couldn't be updated because an INF file contains invalid information.
`0x8024D002`	`WU_E_SETUP_INVALID_IDENTDATA`	Windows Update Agent couldn't be updated because the `wuident.cab` file contains invalid information.
`0x8024D003`	`WU_E_SETUP_ALREADY_INITIALIZED`	Windows Update Agent couldn't be updated because of an internal error that caused setup initialization to be performed twice.
`0x8024D004`	`WU_E_SETUP_NOT_INITIALIZED`	Windows Update Agent couldn't be updated because setup initialization never completed successfully.

Error code	Message	Description
`0x8024D005`	`WU_E_SETUP_SOURCE_VERSION_MISMATCH`	Windows Update Agent couldn't be updated because the versions specified in the INF don't match the actual source file versions.
`0x8024D006`	`WU_E_SETUP_TARGET_VERSION_GREATER`	Windows Update Agent couldn't be updated because a WUA file on the target system is newer than the corresponding source file.
`0x8024D007`	`WU_E_SETUP_REGISTRATION_FAILED`	Windows Update Agent couldn't be updated because `regsvr32.exe` returned an error.
`0x8024D009`	`WU_E_SETUP_SKIP_UPDATE`	An update to the Windows Update Agent was skipped due to a directive in the `wuident.cab` file.
`0x8024D00A`	`WU_E_SETUP_UNSUPPORTED_CONFIGURATION`	Windows Update Agent couldn't be updated because the current system configuration isn't supported.
`0x8024D00B`	`WU_E_SETUP_BLOCKED_CONFIGURATION`	Windows Update Agent couldn't be updated because the system is configured to block the update.
`0x8024D00C`	`WU_E_SETUP_REBOOT_TO_FIX`	Windows Update Agent couldn't be updated because a restart of the system is required.
`0x8024D00D`	`WU_E_SETUP_ALREADYRUNNING`	Windows Update Agent setup is already running.
`0x8024D00E`	`WU_E_SETUP_REBOOTREQUIRED`	Windows Update Agent setup package requires a reboot to complete installation.
`0x8024D00F`	`WU_E_SETUP_HANDLER_EXEC_FAILURE`	Windows Update Agent couldn't be updated because the setup handler failed during execution.
`0x8024D010`	`WU_E_SETUP_INVALID_REGISTRY_DATA`	Windows Update Agent couldn't be updated because the registry contains invalid information.
`0x8024D013`	`WU_E_SETUP_WRONG_SERVER_VERSION`	Windows Update Agent couldn't be updated because the server doesn't contain update information for this version.

Error code	Message	Description
`0x8024DFFF`	`WU_E_SETUP_UNEXPECTED`	Windows Update Agent couldn't be updated because of an error not covered by another `WU_E_SETUP_*` error code.

Feedback

Was this page helpful? Yes No

Provide product feedback

How Windows Update works

Article • 01/19/2023 • Applies to: ✅ Windows 11, ✅ Windows 10

The Windows Update workflow has four core areas of functionality:

1. Scan
   a. Orchestrator schedules the scan.
   b. Orchestrator verifies admin approvals and policies for download.
2. Download
   a. Orchestrator starts downloads.
   b. Windows Update downloads manifest files and provides them to the arbiter.
   c. The arbiter evaluates the manifest and tells the Windows Update client to download files.
   d. Windows Update client downloads files in a temporary folder.
   e. The arbiter stages the downloaded files.
3. Install
   a. Orchestrator starts the installation.
   b. The arbiter calls the installer to install the package.
4. Commit
   a. Orchestrator starts a restart.
   b. The arbiter finalizes before the restart.

How updating works

During the updating process, the Windows Update Orchestrator operates in the background to scan, download, and install updates. It does these actions automatically, according to your settings, and silently so that doesn't disrupt your computer usage.

Scanning updates

The Windows Update Orchestrator on your PC checks the Microsoft Update server or your WSUS endpoint for new updates at random intervals. The randomization ensures that the Windows Update server isn't overloaded with requests all at the same time. The

Update Orchestrator searches only for updates that have been added since the last time updates were searched, allowing it to find updates quickly and efficiently.

When devices check for updates, the Windows Update Orchestrator evaluates whether the update is appropriate for your device. It uses guidelines defined by the publisher of the update, for example, Microsoft Office including enterprise group policies.

Make sure you're familiar with the following terminology related to Windows Update scan:

Expand table

Term	Definition
Update	We use this term to mean several different things, but in this context it's the actual updated code or change.
Bundle update	An update that contains 1-N child updates; doesn't contain payload itself.
Child update	Leaf update that's bundled by another update; contains payload.
Detector update	A special update that contains `IsInstalled` applicability rule only and no payload. Used for prerequisite evaluation.
Category update	A special `detectoid` that has an `IsInstalled` rule that is always true. Used for grouping updates and allowing the device to filter updates.
Full scan	Scan with empty datastore.
Delta scan	Scan with updates from previous scan already cached in datastore.
Online scan	Scan that uses the network and to check an update server.
Offline scan	Scan that doesn't use the network and instead checks the local datastore. Only useful if online scan has been performed before.
CatScan	Category scan where caller can specify a **categoryId** to get updates published under that **categoryId**.
AppCatScan	Category scan where caller can specify an **AppCategoryId** to get apps published under that **appCategoryId**.
Software sync	Part of the scan that only checks for software updates (both the apps and the operating system).
Driver sync	Part of the scan that checks driver updates only. This sync is optional and runs after the software sync.
ProductSync	A sync based on attributes, in which the client provides a list of device, product, and caller attributes ahead of time to allow service to check applicability in the

Term	Definition
	cloud.

How Windows Update scanning works

Windows Update does the following actions when it runs a scan.

Starts the scan for updates

When users start scanning in Windows Update through the Settings panel, the following occurs:

- The scan first generates a `ComApi` message. The caller (Microsoft Defender Antivirus) tells the Windows Update engine to scan for updates.
- "Agent" messages: queueing the scan, then actually starting the work:
  - Updates are identified by the different IDs ("ID = 10", "ID = 11") and from the different thread ID numbers.
  - Windows Update uses the thread ID filtering to concentrate on one particular task.

```
05/26/2017-11:30:25.982 0A1C 41F0 ComApi    * START *   Search ClientId = Windows Defender Antivirus (77BDAF73-B396-481F-9042-AD358843EC24), ServiceId = 7971F918-A847-4430-9279-4A52D1EFE18D (cV = IGqGElKdWEyr9okh.1.0.0)
05/26/2017-11:30:26.037 3A68 4A14 Agent     * START * Queueing Finding updates [CallerId = Windows Defender Antivirus (77BDAF73-B396-481F-9042-AD358843EC24)  Id = 10]
05/26/2017-11:30:26.037 0A1C 41F0 ComApi    * START *   Search ClientId = Windows Defender Antivirus (77BDAF73-B396-481F-9042-AD358843EC24), ServiceId = 8B24B027-1DEE-BABB-9A95-3517DFB9C552 (cV = IGqGElKdWEyr9okh.1.0.0)
05/26/2017-11:30:26.045 3A68 4918 Agent     * END * Queueing Finding updates [CallerId = Windows Defender Antivirus (77BDAF73-B396-481F-9042-AD358843EC24)  Id = 10]
05/26/2017-11:30:26.052 3A68 4918 Agent     * START * Finding updates CallerId = Windows Defender Antivirus (77BDAF73-B396-481F-9042-AD358843EC24)  Id = 10
05/26/2017-11:30:26.052 3A68 4918 Agent     Criteria = "(IsInstalled = 0 and IsHidden = 0 and CategoryIDs contains '8c3fcc84-7410-4a95-8b89-a166a0190486' and CategoryIDs contains 'e0789628-ce08-4437-be74-2495b842f43b')"
05/26/2017-11:30:26.052 3A68 4918 Agent     ServiceID = {7971F918-A847-4430-9279-4A52D1EFE18D} Third party service
05/26/2017-11:30:26.062 3A68 4A14 Agent     * START * Queueing Finding updates [CallerId = Windows Defender Antivirus (77BDAF73-B396-481F-9042-AD358843EC24)  Id = 11]
05/26/2017-11:30:26.070 3A68 39F8 Agent     * END * Queueing Finding updates [CallerId = Windows Defender Antivirus (77BDAF73-B396-481F-9042-AD358843EC24)  Id = 11]
05/26/2017-11:30:26.077 3A68 39F8 Agent     * START * Finding updates CallerId = Windows Defender Antivirus (77BDAF73-B396-481F-9042-AD358843EC24)  Id = 11
05/26/2017-11:30:26.077 3A68 39F8 Agent     Criteria = "(IsInstalled = 0 and IsHidden = 0 and CategoryIDs contains '8c3fcc84-7410-4a95-8b89-a166a0190486' and CategoryIDs contains 'e0789628-ce08-4437-be74-2495b842f43b')"
05/26/2017-11:30:26.077 3A68 39F8 Agent     ServiceID = {8B24B027-1DEE-BABB-9A95-3517DFB9C552} Third party service
```

Proxy Behavior

For Windows Update (WU) scans URLs that are used for update detection (MS-WUSP: SimpleAuth Web Service, MS-WUSP: Client Web Service):

- System proxy is attempted (set using the `netsh` command).
- If WUA fails to reach the service due to a certain proxy, service, or authentication error code, then user proxy is attempted (generally it's the logged-in user).

> **Note**
>
> For intranet WSUS update service URLs, we provide an option via Windows Update policy to select the proxy behavior.

For Windows Update URLs that *aren't* used for update detection, such as for download or reporting:

- User proxy is attempted.
- If WUA fails to reach the service due to a certain proxy, service, or authentication error code, then the system proxy is attempted.

Identifies service IDs

- Service IDs indicate which update source is being scanned.
- The Windows Update engine treats every service as a separate entity, even though multiple services may contain the same updates.

```
05/26/2017-11:30:25.982 0A1C 41F0 ComApi    * START *   Search ClientId = Windows Defender Antivirus (77BDAF73-B396-481F-9042-AD358843EC24), ServiceId = 7971F918-A847-4430-9279-4A52D1EFE18D (cV = IGqGEIKdWEyr9okh.1.0.0)
05/26/2017-11:30:26.037 3A68 4A14 Agent     * START * Queueing Finding updates [CallerId = Windows Defender Antivirus (77BDAF73-B396-481F-9042-AD358843EC24)  Id = 10]
05/26/2017-11:30:26.037 0A1C 41F0 ComApi    * START *   Search ClientId = Windows Defender Antivirus (77BDAF73-B396-481F-9042-AD358843EC24), ServiceId = 8B24B027-1DEE-BABB-9A95-3517DFB9C552 (cV = IGqGEIKdWEyr9okh.1.0.0)
05/26/2017-11:30:26.045 3A68 4918 Agent     * END * Queueing Finding updates [CallerId = Windows Defender Antivirus (77BDAF73-B396-481F-9042-AD358843EC24)  Id = 10]
05/26/2017-11:30:26.052 3A68 4918 Agent     * START * Finding updates CallerId = Windows Defender Antivirus (77BDAF73-B396-481F-9042-AD358843EC24)  Id = 10
05/26/2017-11:30:26.052 3A68 4918 Agent     Criteria = "(IsInstalled = 0 and IsHidden = 0 and CategoryIDs contains '8c3fcc84-7410-4a95-8b89-a166a0190486' and CategoryIDs contains 'e0789628-ce08-4437-be74-2495b842f43b')"
05/26/2017-11:30:26.052 3A68 4918 Agent     ServiceID = {7971F918-A847-4430-9279-4A52D1EFE18D} Third party service
05/26/2017-11:30:26.062 3A68 4A14 Agent     * START * Queueing Finding updates [CallerId = Windows Defender Antivirus (77BDAF73-B396-481F-9042-AD358843EC24)  Id = 11]
05/26/2017-11:30:26.070 3A68 39F8 Agent     * END * Queueing Finding updates [CallerId = Windows Defender Antivirus (77BDAF73-B396-481F-9042-AD358843EC24)  Id = 11]
05/26/2017-11:30:26.077 3A68 39F8 Agent     * START * Finding updates CallerId = Windows Defender Antivirus (77BDAF73-B396-481F-9042-AD358843EC24)  Id = 11
05/26/2017-11:30:26.077 3A68 39F8 Agent     Criteria = "(IsInstalled = 0 and IsHidden = 0 and CategoryIDs contains '8c3fcc84-7410-4a95-8b89-a166a0190486' and CategoryIDs contains 'e0789628-ce08-4437-be74-2495b842f43b')"
05/26/2017-11:30:26.077 3A68 39F8 Agent     ServiceID = {8B24B027-1DEE-BABB-9A95-3517DFB9C552} Third party service
```

- Common service IDs

> **Important**
>
> ServiceId here identifies a client abstraction, not any specific service in the cloud. No assumption should be made of which server a serviceId is pointing to. It's totally controlled by responses from the Service Locator Service.

Expand table

Service	ServiceId
Unspecified / Default	Windows Update, Microsoft Update, or WSUS 00000000-0000-0000-0000-000000000000
Windows Update	9482F4B4-E343-43B6-B170-9A65BC822C77
Microsoft Update	7971f918-a847-4430-9279-4a52d1efe18d
Store	855E8A7C-ECB4-4CA3-B045-1DFA50104289
OS Flighting	8B24B027-1DEE-BABB-9A95-3517DFB9C552
WSUS or Configuration Manager	Via ServerSelection::ssManagedServer 3DA21691-E39D-4da6-8A4B-B43877BCB1B7
Offline scan service	Via IUpdateServiceManager::AddScanPackageService

Finds network faults

Common update failure is caused due to network issues. To find the root of the issue:

- Look for "ProtocolTalker" messages to see client-server sync network traffic.
- "SOAP faults" can be either client- or server-side issues; read the message.
- The Windows Update client uses the Service Locator Service to discover the configurations and endpoints of Microsoft network update sources: Windows update, Microsoft Update, or Flighting.

> **Note**
>
> If the search is against WSUS or Configuration Manager, you can ignore warning messages for the Service Locator Service.

- On sites that only use WSUS or Configuration Manager, the Service Locator Service might be blocked at the firewall. In this case the request will fail, and though the service can't scan against Windows Update or Microsoft Update, it can still scan against WSUS or Configuration Manager, since it's locally configured.

```
5/26/2017-17:30:38:38.8383104 7368  1504  ProtocolTalker  [0]1CC8.05E0::06/22/2017-22:02:38.838 [agent]ServiceId = {8B24B027-1DEE-BABB-9A95-3517DFB9C552}, Server URL = https://fe3.delivery.mp.microsoft.com/ClientWebService/client.asmx
5/26/2017-17:30:38:38.8383129 7368  1504  ProtocolTalker  [0]1CC8.05E0::06/22/2017-22:02:38.838 [agent]OK to reuse existing configuration
5/26/2017-17:30:38:38.8383166 7368  1504  ProtocolTalker  [0]1CC8.05E0::06/22/2017-22:02:38.838 [agent]Existing cookie is valid, just use it
```

Downloading updates

Once the Windows Update Orchestrator determines which updates apply to your computer, it begins downloading the updates, if you have selected the option to automatically download updates. It does operation in the background without interrupting your normal use of the device.

To ensure that your other downloads aren't affected or slowed down because updates are downloading, Windows Update uses Delivery Optimization, which downloads updates and reduces bandwidth consumption.

For more information, see Configure Delivery Optimization for Windows 10 updates.

Installing updates

When an update is applicable, the "Arbiter" and metadata are downloaded. Depending on your Windows Update settings, when downloading is complete, the Arbiter will gather details from the device, and compare that with the downloaded metadata to create an "action list".

The action list describes all the files needed from Windows Update, and what the installation agent (such as CBS or Setup) should do with them. The action list is provided to the installation agent along with the payload to begin the installation.

Committing Updates

When the option to automatically install updates is configured, the Windows Update Orchestrator, in most cases, automatically restarts the device for you after installing the updates. It has to restart the device because it might be insecure, or not fully updated, until it restarts. You can use Group Policy settings, mobile device management (MDM), or the registry (not recommended) to configure when devices will restart after a Windows 10 update is installed.

For more information, see Manage device restarts after updates.

Feedback

Was this page helpful? Yes No

Provide product feedback

Windows upgrade paths

Applies to: ✅ Windows 10, ✅ Windows 11

Upgrade paths

This article provides a summary of available upgrade paths to currently supported versions of Windows. You can upgrade to currently supported versions of Windows from previous versions of Windows that are also still supported. Paths include upgrading from one release of Windows to a later release of the same version of Windows. Migrating from one edition of Windows to a different edition of the same release is also supported.

> **Note**
>
> If you're also migrating to a different edition of Windows, see Windows edition upgrade. The Windows edition upgrade article describes methods and supported paths to change the edition of Windows. These methods require that you input a license or product key for the new Windows edition prior to starting the upgrade process. Edition downgrade is also supported for some paths. However, applications and settings aren't always maintained when the Windows edition is downgraded.

- **Windows version upgrade**: You can directly upgrade any General Availability Channel version of Windows to a newer, supported General Availability Channel version of Windows, even if it involves skipping versions. Work with your account representative if your current version of Windows is out of support. See the Windows lifecycle fact sheet for availability and service information.

- **Upgrade from Windows LTSC to Windows General Availability Channel**: Upgrade from Windows LTSC to Windows General Availability Channel is available when upgrading to the same or a newer build version. For example, Windows 10 Enterprise 2016 LTSB can be upgraded to Windows 10 Enterprise 22H2. Upgrade is supported using the in-place upgrade process using Windows setup. The Product Key switch needs to be used if apps need to be kept. If the switch isn't used, the option **Keep personal files and apps** option is grayed out. The command line to perform the upgrade is:

  Windows Command Prompt

  ```
  setup.exe /pkey xxxxx-xxxxx-xxxxx-xxxxx-xxxxx
  ```

  where **xxxxx-xxxxx-xxxxx-xxxxx-xxxxx** is the Windows General Availability Channel product key. For example, if using a KMS, the command line for Windows Enterprise

would be:

Windows Command Prompt

```
setup.exe /pkey NPPR9-FWDCX-D2C8J-H872K-2YT43
```

For additional product keys for use with KMS, see Key Management Services (KMS) client activation and product keys:Generic Volume License Keys (GVLK).

> **Important**
>
> In-place upgrade from Windows General Availability Channel to Windows LTSC isn't supported.
>
> Windows 10 LTSC 2015 didn't block this in-place upgrade path even though it isn't supported. This issue was corrected in the Windows 10 LTSC 2016 release. Windows 10 LTSC 2016 only allows data-only and clean install options.

- **Windows N/KN**: **Windows N** and **Windows KN** SKUs (editions without media-related functionality) follow the same upgrade paths shown in the following tables. If the pre-upgrade and post-upgrade editions aren't the same type, for example, Windows 10 Pro N to Windows 11 Pro, personal data is kept but applications and settings are removed during the upgrade process.

Supported Windows upgrade paths

Expand table

Windows Edition	**Upgrade to** Windows Home	**Upgrade to** Windows Pro	**Upgrade to** Windows Pro Education	**Upgrade to** Windows Education	**Upgrade to** Windows Enterprise
Upgrade from Windows Home	❌	✅	✅	✅	❌
Upgrade from Windows Pro	⬇️	❌	✅	✅	✅
Upgrade from Windows Education	❌	❌	❌	❌	⬇️

Windows Edition	Upgrade to Windows Home	Upgrade to Windows Pro	Upgrade to Windows Pro Education	Upgrade to Windows Education	Upgrade to Windows Enterprise
Upgrade from Windows Enterprise	❌	❌	❌	✅	❌

- ✅ = Full upgrade is supported including personal data, settings, and applications.
- ❌ = Upgrade isn't supported or not applicable.
- ⬇️ = Edition downgrade; personal data is maintained, applications and settings are removed.

Related articles

- Windows 10 deployment scenarios.
- Windows upgrade and migration considerations.
- Windows edition upgrade.

Last updated on 04/24/2025

Windows edition upgrade

Article • 10/04/2024 • Applies to: ✅ Windows 10, ✅ Windows 11

With Windows, you can quickly upgrade from one edition of Windows to another, provided the upgrade path is supported. For information on what edition of Windows is right for you, see the following articles:

- Compare Windows 11 Editions.
- Explore Windows 11 Pro features.
- Windows 10 Pro vs Windows 11 Pro.
- Compare Windows 10 Editions.
- Windows For Business.

For a comprehensive list of all possible upgrade paths to Windows, see Windows upgrade paths. Downgrading the edition of Windows is discussed in the License expiration section in this article.

The following table shows the methods and paths available to change the edition of Windows that is running on your computer.

Expand table

Edition upgrade	MDM	Provisioning package	Command-line tool	Manually entering product key
Home > Pro	❌	❌	❌	☑
Home > Pro for Workstations	❌	❌	❌	☑
Home > Pro Education	☑	☑	☑	☑
Home > Education	☑	☑	☑	☑
Pro > Pro for Workstations	✅	✅	✅	✅
Pro > Pro Education	✅	✅	✅	✅
Pro > Education	☑	☑	☑	☑
Pro > Enterprise	✅	✅	✅	✅
Pro for Workstations > Pro Education	✅	✅	✅	✅
Pro for Workstations > Education	☑	☑	☑	☑

Edition upgrade	MDM	Provisioning package	Command-line tool	Manually entering product key
Pro for Workstations > Enterprise	✅	✅	✅	✅
Pro Education > Education	☑️	☑️	☑️	☑️
Enterprise > Education	☑️	☑️	☑️	☑️

- ✅ = Supported, no reboot required.
- ☑️ = Supported, but reboot required.
- ❌ = Not supported.
- MDM = Modern device management.

> **Note**
>
> - Each desktop edition in the table also has an N and KN SKU. These editions have had media-related functionality removed. Devices with N or KN SKUs installed can be upgraded to corresponding N or KN SKUs using the same methods.
> - Edition upgrades via Microsoft Store for Business are no longer available with the retirement of the Microsoft Store for Business. For more information, see Microsoft Store for Business and Education retiring March 31, 2023.

> **Tip**
>
> Edition upgrade is also possible using edition upgrade policy in Microsoft Configuration Manager. For more information, see Upgrade Windows devices to a new edition with Configuration Manager.

Upgrade using modern device management (MDM)

To upgrade desktop editions of Windows using MDM, enter the product key for the upgraded edition in the **UpgradeEditionWithProductKey** policy setting of the **WindowsLicensing** CSP. For more info, see WindowsLicensing CSP.

For information on upgrading editions of Windows using Microsoft Intune, including switching out of S mode, see Upgrade Windows 10/11 editions or switch out of S mode on devices using Microsoft Intune.

Upgrade using a provisioning package

Use Windows Configuration Designer to create a provisioning package to upgrade a desktop edition of Windows. Windows Configuration Designer is available as part of the Windows Assessment and Deployment Kit (Windows ADK) or as a stand-alone Microsoft Store app. Download the Windows Configuration Designer from one of the following locations:

- Download and install the Windows ADK - When installing the ADK, make sure to select **Configuration Designer**. After installation, Windows Configuration Designer can be found in the Start Menu under **Windows Kits** > **Windows Imaging and Configuration Designer**.
- Windows Configuration Designer - Microsoft Store app. After installation, Windows Configuration Designer can be found in the Start menu as **Windows Configuration Designer**.

  Download Windows Configuration Designer from the Microsoft Store

To create a provisioning package for upgrading desktop editions of Windows:

1. Open Windows Configuration Designer.
2. Select **Advanced provisioning**.
3. In the **New project** window that opens:
   a. Under **Enter project details**, give the project a name and then select the **Next** button.
   b. Under **Choose which settings to view and configure**, select **All Windows desktop editions** and then select the **Next** button.
   c. Under **Import a provisioning package (optional)**, select the **Finish** button.
4. Under **Available customizations**, expand **Runtime settings** > **EditionUpgrade** and then select **ChangeProductKey**.
5. In the **EditionUpgrade/ChangeProductKey** pane, next to **ChangeProductKey**, enter the product key for the upgraded edition.

6. Under the **File** menu, select **Save**.

7. Under the **Export** menu, select **Provisioning package**.

8. Step through the **Build** wizard to create the provisioning package, making sure to save the provisioning package to a known location.

For more info about Windows Configuration Designer, see the following articles:

- Create a provisioning package
- Apply a provisioning package

Upgrade using a command-line tool

The `changepk.exe` command-line tool can be used to upgrade devices to a supported edition of Windows:

Windows Command Prompt

```
changepk.exe /ProductKey <product_key>`
```

Upgrades can also be performed using `slmgr.vbs` and a KMS client setup key. For example:

Windows Command Prompt

```
cscript.exe c:\windows\system32\slmgr.vbs /ipk <product_key>
```

Upgrade by manually entering a product key

If only a few devices are being upgraded devices, a product key for the upgraded edition can be entered manually. To manually enter a product key:

1. Right click on the **Start** menu and select **Run**.

2. In the **Run** window, next to **Open**, enter

   `ms-settings:activation`

   and then select **OK**.

3. Select **Change product key**.

4. Enter your product key.

5. Follow the on-screen instructions.

Alternatively, select the following link to automatically open the **Settings** app to the activation page:

Activation

Upgrade by purchasing a license from the Microsoft Store

If you don't have a product key, you can upgrade your edition of Windows through the Microsoft Store. To upgrade through the Microsoft Store:

1. Right click on the **Start** menu and select **Run**.
2. In the **Run** window, next to **Open**, enter

   `ms-windows-store://windowsupgrade/`

   and then select **OK**.
3. Follow the on-screen instructions.

Alternatively, select the following link to automatically open the Microsoft Store to the page for upgrading the edition of Windows:

Upgrade Windows Edition

License expiration

Volume license customers whose license has expired need to change the edition of Windows to an edition with an active license. Downgrading the edition of Windows is possible using the same methods that were used to perform an edition upgrade. If the downgrade path is supported, then apps and settings can be migrated from the current edition. If a path isn't supported, then a clean install is required.

The following scenarios aren't supported:

- Downgrading Windows to a pervious version by entering a different product key, for example from Windows 11 Pro to Windows 10 Pro.
- Downgrading from a later version to an earlier version of the same edition of Windows, for example from Windows 11 Pro 22H2 to Windows 11 Pro 22H1,

unless using rollback.

> **Note**
>
> If you're using Windows subscription activation and a license expires, devices automatically revert to the original edition when the grace period expires.

Supported Windows downgrade paths

Expand table

Edition	Home	Pro	Pro for Workstations	Pro Education	Education	Enterprise LTSC	Enterprise
Home	-	❌	❌	❌	❌	❌	❌
Pro	❌	-	❌	❌	❌	❌	❌
Pro for Workstations	❌	❌	-	❌	❌	❌	❌
Pro Education	❌	❌	❌	-	❌	❌	❌
Education	❌	✅	✅	✅	-	❌	-
Enterprise LTSC	❌	❌	❌	❌	❌	-	❌
Enterprise	❌	✅	✅	✅	-	❌	-

- ✅ = Supported downgrade path.
- ❌ = not supported or not a downgrade.
- - = Not considered a downgrade or an upgrade.

> **Note**
>
> Windows **N** and Windows **KN** SKUs follow the same rules shown in the table.

The table may not represent more complex scenarios. For example, you can perform an upgrade from Pro to Pro for Workstation on a computer with an embedded Pro key using a Pro for Workstation license key. You can then later downgrade this computer back to Pro with the firmware-embedded key. The downgrade is allowed but only because the pre-installed OS is Pro.

Scenario example: Downgrading from Enterprise

- Original edition: **Professional OEM**
- Upgrade edition: **Enterprise**
- Valid downgrade paths: **Pro, Pro for Workstations, Pro Education, Education**

You can move directly from Enterprise to any valid destination edition. In this example, downgrading to Pro for Workstations, Pro Education, or Education requires an additional activation key to supersede the firmware-embedded Pro key. In all cases, you must comply with Microsoft License Terms. If you're a volume license customer, refer to the Microsoft Volume Licensing Reference Guide.

Related articles

- Windows upgrade paths
- Volume Licensing Service Center
- Windows Subscription Activation

Feedback

Was this page helpful? Yes No

Provide product feedback

Deploy Windows with Microsoft 365

Article • 02/27/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

This article provides a brief overview of Microsoft 365 and describes how to use a free 90-day trial account to review some of the benefits of Microsoft 365.

Microsoft 365 is an offering from Microsoft that combines Windows with Office 365, and Enterprise Mobility and Security (EMS). See the Microsoft 365 Enterprise poster for an overview.

For Windows deployment, Microsoft 365 includes a deployment advisor that walks through the entire process of deploying Windows. The wizard supports multiple Windows deployment methods, including:

- Windows Autopilot.
- In-place upgrade.
- Deploying Windows upgrade with Intune.
- Deploying Windows upgrade with Microsoft Configuration Manager.
- Deploying a computer refresh with Microsoft Configuration Manager.

Free trial account

If an existing Microsoft services subscription account exists, and there's access to the Microsoft 365 Admin Center:

1. Sign into the Microsoft 365 Admin Center.
2. Go to **Billing** and then **Purchase services**.
3. In the Enterprise Suites section of the service offerings, find the Microsoft 365 E3 and Microsoft 365 E5 tiles.
4. Select one of the available **Start Free Trial** options.

If there isn't an existing Microsoft services subscription, Microsoft 365 deployment advisor and other resources can be tried for free! Just follow these steps:

1. Obtain a free Microsoft 365 trial.
2. Check out the Microsoft 365 deployment advisor.

> **Note**
>
> When setup guide runs for the first time, the **Prepare your environment** guide appears. This guide makes sure the basics are covered like domain verification and a method for adding users. At the end of the **Prepare your environment** guide,

there's a **Ready to continue** button that goes back to the original guide that was selected.

Microsoft 365 Enterprise poster

Select Microsoft 365 Enterprise poster to see the latest version of the Microsoft 365 Enterprise poster.

Related articles

- Windows deployment scenarios.

Feedback

Was this page helpful? Yes No

Provide product feedback

Get started with Windows Update

Article • 08/28/2023 • Applies to: ✅ Windows 11, ✅ Windows 10

> Applies to: Windows 10

With the release of Windows 10, we moved the update model to the Unified Update Platform. Unified Update Platform (UUP) is a single publishing, hosting, scan and download model for all types of OS updates, client devices for all Windows-based operating systems, for everything from monthly quality updates to new feature updates.

Use the following information to get started with Windows Update:

- Understand the UUP architecture
- Understand how Windows Update works
- Find Windows Update log files
- Learn how to troubleshoot Windows Update
- Review common Windows Update errors and check out the error code reference
- Review other resources to help you use Windows Update
- Review Windows IT Pro Blog section of Microsoft Blogs.

Unified Update Platform (UUP) architecture

To understand the changes to the Windows Update architecture that UUP introduces let's start with some new key terms.

Unified Update Platform	Client Upgrades	Client Updates
User Experience	Update UI	
Update Scheduling	Orchestrator	
Check for Updates and Downloads	WU Client	
Install Handler	WU Arbiter Handler	
Deployment Manager	Deployment Arbiter	
Installer	Upgrade Engine	CBS

- **Update UI** - The user interface to initiate Windows Update check and history. Available under **Settings --> Update & Security --> Windows Update**.

- **Update Session Orchestrator (USO)**- A Windows OS component that orchestrates the sequence of downloading and installing various update types from Windows Update.

  Update types-

  - OS Feature updates
  - OS Security updates
  - Device drivers
  - Defender definition updates

  > ⓘ **Note**
  >
  > Other types of updates, like Office desktop updates, are installed if the user opts into Microsoft Update.
  >
  > Store apps aren't installed by USO, today they are separate.

- **Windows Update Client/ UpdateAgent** - The component running on your PC. It's essentially a DLL that is downloaded to the device when an update is applicable. It surfaces the APIs needed to perform an update, including those needed to generate a list of payloads to download, as well as starts stage and commit operations. It provides a unified interface that abstracts away the underlying update technologies from the caller.

- **Windows Update Arbiter handle**- Code that is included in the UpdateAgent binary. The arbiter gathers information about the device, and uses the CompDB(s) to output an action list. It is responsible for determining the final "composition state" of your device, and which payloads (like ESDs or packages) are needed to get your device up to date.

- **Deployment Arbiter**- A deployment manager that calls different installers. For example, CBS.

Additional components include the following-

- **CompDB** - A generic term to refer to the XML describing information about target build composition, available diff packages, and conditional rules.
- **Action List** - The payload and additional information needed to perform an update. The action list is consumed by the UpdateAgent, as well as other installers to determine what payload to download. It's also consumed by the "Install Agent"

to determine what actions need to be taken, such as installing or removing packages.

Feedback

Was this page helpful? Yes No

Provide product feedback

Windows Update log files

Article • 12/08/2023 • Applies to: ✅ Windows 11, ✅ Windows 10

The following table describes the log files created by Windows Update.

Expand table

Log file	Location	Description	When to use
windowsupdate.log	C:\Windows\Logs\WindowsUpdate	Starting in Windows 8.1 and continuing in Windows 10, Windows Update client uses Event Tracing for Windows (ETW) to generate diagnostic logs.	If you receive an error message when you run Windows Update, you can use the information included in the Windowsupdate.log log file to troubleshoot the issue.
UpdateSessionOrchestration.etl	C:\ProgramData\USOShared\Logs	Starting Windows 10, the Update Orchestrator Service is responsible for sequence of downloading and installing various update types from Windows Update. And the events are logged to these .etl files.	• When you see that the updates are available but download isn't getting triggered. • When updates are downloaded but installation isn't triggered. • When updates are installed but reboot isn't triggered.
NotificationUxBroker.etl	C:\ProgramData\USOShared\Logs	Starting Windows 10, the notification toast or the banner is triggered by NotificationUxBroker.exe.	When you want to check whether the notification was triggered or not.
CBS.log	%systemroot%\Logs\CBS	This log provides insight on the update installation part in the servicing stack.	To troubleshoot the issues related to Windows Update installation.

Generating WindowsUpdate.log

To merge and convert Windows Update trace files (.etl files) into a single readable WindowsUpdate.log file, see Get-WindowsUpdateLog.

> ⓘ **Note**
>
> When you run the **Get-WindowsUpdateLog** cmdlet, an copy of WindowsUpdate.log file is created as a static log file. It does not update as the old WindowsUpdate.log unless you run **Get-WindowsUpdateLog** again.

Windows Update log components

The Windows Update engine has different component names. The following are some of the most common components that appear in the WindowsUpdate.log file:

- AGENT- Windows Update agent
- AU - Automatic Updates is performing this task
- AUCLNT- Interaction between AU and the logged-on user
- CDM- Device Manager
- CMPRESS- Compression agent
- COMAPI- Windows Update API
- DRIVER- Device driver information
- DTASTOR- Handles database transactions
- EEHNDLER- Expression handler that's used to evaluate update applicability
- HANDLER- Manages the update installers
- MISC- General service information
- OFFLSNC- Detects available updates without network connection
- PARSER- Parses expression information
- PT- Synchronizes updates information to the local datastore
- REPORT- Collects reporting information
- SERVICE- Startup/shutdown of the Automatic Updates service
- SETUP- Installs new versions of the Windows Update client when it's available
- SHUTDWN- Install at shutdown feature
- WUREDIR- The Windows Update redirector files
- WUWEB- The Windows Update ActiveX control
- ProtocolTalker - Client-server sync
- DownloadManager - Creates and monitors payload downloads
- Handler, Setup - Installer handlers (CBS, and so on)
- EEHandler - Evaluating update applicability rules
- DataStore - Caching update data locally
- IdleTimer - Tracking active calls, stopping a service

> ⓘ **Note**

Many component log messages are invaluable if you are looking for problems in that specific area. However, they can be useless if you don't filter to exclude irrelevant components so that you can focus on what's important.

Windows Update log structure

The Windows update log structure is separated into four main identities:

- Time stamps
- Process ID and thread ID
- Component name
- Update identifiers
  - Update ID and revision number
  - Revision ID
  - Local ID
  - Inconsistent terminology

The WindowsUpdate.log structure is discussed in the following sections.

Time stamps

The time stamp indicates the time at which the logging occurs.

- Messages are usually in chronological order, but there may be exceptions.
- A pause during a sync can indicate a network problem, even if the scan succeeds.
- A long pause near the end of a scan can indicate a supersedence chain issue.

Process ID and thread ID

The Process IDs and Thread IDs are random, and they can vary from log to log and even from service session to service session within the same log.

- The first four digits, in hex, are the process ID.
- The next four digits, in hex, are the thread ID.
- Each component, such as the USO, Windows Update engine, COM API callers, and Windows Update installer handlers, has its own process ID.

```
05/26/2017-11:30:26.07  3A68 39F8  gent   Criteria = "(IsInstalled = 0 and IsHidden = 0 and C
05/26/2017-11:30:26.07  3A68 39F8  gent   ServiceID = {8B24B027-1DEE-BABB-9A95-3517
05/26/2017-11:30:26.15  3A68 4918  otocolTalker   ServiceId = {7971F918-A847-4430-927
05/26/2017-11:30:26.36  3A68 4918  otocolTalker   ServiceId = {7971F918-A847-4430-927
05/26/2017-11:30:26.42  3A68 4918  gent   update {9E4D4E24-B06F-4930-BC70-77C563FC
05/26/2017-11:30:26.42  3A68 4918  gent   update {79D6195D-89A6-4391-B550-EC5A7165
05/26/2017-11:30:26.42  3A68 4918  gent   update {CAF1D4A2-0881-46D0-8552-170349A1
05/26/2017-11:30:26.42  3A68 4918  gent   update {DE25AECF-4BF6-4094-98C3-7896860D
05/26/2017-11:30:26.42  3A68 4918  gent   update {AD2BC8ED-5A55-4C2F-A891-F886FE2
05/26/2017-11:30:26.42  3A68 4918  gent   update {228A2AF3-2B2A-4A32-AEA6-1BA26C92
```

Component name

Search for and identify the components that are associated with the IDs. Different parts of the Windows Update engine have different component names. Some of them are as follows:

- ProtocolTalker - Client-server sync
- DownloadManager - Creates and monitors payload downloads
- Handler, Setup - Installer handlers (CBS, etc.)
- EEHandler - Evaluating update applicability rules
- DataStore - Caching update data locally
- IdleTimer - Tracking active calls, stopping service

```
05/26/2017-11:30:26.077 3A68 39F  Agent   Criteria = "(IsInstalled = 0 and IsHidden = 0 and C
05/26/2017-11:30:26.077 3A68 39F  Agent   Service  = {8B24B027-1DEE-BABB-9A95-3517
05/26/2017-11:30:26.159 3A68 491  ProtocolTalker   erviceId = {7971F918-A847-4430-927
05/26/2017-11:30:26.363 3A68 491  ProtocolTalker   erviceId = {7971F918-A847-4430-927
05/26/2017-11:30:26.428 3A68 491  Agent   update  E4D4E24-B06F-4930-BC70-77C563FC
05/26/2017-11:30:26.428 3A68 491  Agent   update  9D6195D-89A6-4391-B550-EC5A7165
05/26/2017-11:30:26.428 3A68 491  Agent   update  AF1D4A2-0881-46D0-8552-170349A1
05/26/2017-11:30:26.428 3A68 491  Agent   update  E25AECF-4BF6-4094-98C3-7896860D
05/26/2017-11:30:26.428 3A68 491  Agent   update  D2BC8ED-5A55-4C2F-A891-F886FE2
05/26/2017-11:30:26.428 3A68 491  Agent   update  28A2AF3-2B2A-4A32-AEA6-1BA26C92
```

Update identifiers

The following items are update identifiers:

Update ID and revision number

There are different identifiers for the same update in different contexts. It's important to know the identifier schemes.

- Update ID: A GUID (indicated in the previous screenshot) assigned to a given update at publication time
- Revision number: A number incremented every time that a given update (that has a given update ID) is modified and republished on a service
- Revision numbers are reused from one update to another (not a unique identifier).
- The update ID and revision number are often shown together as "{GUID}.revision."

```
05/26/2017-11:30:26.428 3A68 4918 Agent   updat {9E4D4E24-B06F-4930-BC70-77C563FC8C48}.200 is
```

Revision ID

- A Revision ID (don't confuse this value with "revision number") is a serial number issued when an update is initially published or revised on a given service.
- An existing update that is revised keeps the same update ID (GUID), has its revision number incremented (for example, from 100 to 101), but gets a new revision ID that isn't related to the previous ID.
- Revision IDs are unique on a given update source, but not across multiple sources.
- The same update revision might have different revision IDs on Windows Update and WSUS.
- The same revision ID might represent different updates on Windows Update and WSUS.

Local ID

- Local ID is a serial number issued by a given Windows Update client when an update is received from a service.
- Typically seen in debug logs, especially involving the local cache for update info (Datastore)
- Different client PCs assign different Local IDs to the same update
- You can find the local IDs that a client is using by getting the client's %WINDIR%\SoftwareDistribution\Datastore\Datastore.edb file

Inconsistent terminology

- Sometimes the logs use terms inconsistently. For example, the InstalledNonLeafUpdateIDs list actually contains revision IDs, not update IDs.

- Recognize IDs by form and context:
  - GUIDs are update IDs
  - Small integers that appear alongside an update ID are revision numbers
  - Large integers are typically revision IDs
  - Small integers (especially in Datastore) can be local IDs

```
05/26/2017-11:30:27.101 3A68 39F8 ProtocolTalker   SyncUpdateParameters (Loop 1):
05/26/2017-11:30:27.101 3A68 39F8 ProtocolTalker     ExpressQuery: FALSE
05/26/2017-11:30:27.101 3A68 39F8 ProtocolTalker     InstalledNonLeafUpdateIDs (22):
05/26/2017-11:30:27.101 3A68 39F8 ProtocolTalker       1
05/26/2017-11:30:27.101 3A68 39F8 ProtocolTalker       2
05/26/2017-11:30:27.101 3A68 39F8 ProtocolTalker       3
05/26/2017-11:30:27.101 3A68 39F8 ProtocolTalker       5169044
05/26/2017-11:30:27.101 3A68 39F8 ProtocolTalker       23110993
05/26/2017-11:30:27.101 3A68 39F8 ProtocolTalker       59830006
05/26/2017-11:30:27.101 3A68 39F8 ProtocolTalker       60484010
05/26/2017-11:30:27.101 3A68 39F8 ProtocolTalker       62450019
05/26/2017-11:30:27.101 3A68 39F8 ProtocolTalker       98959023
```

Windows Setup log files analysis using SetupDiag tool

SetupDiag is a diagnostic tool that can be used for analysis of logs related to installation of Windows Updates. For more information, see SetupDiag.

Feedback

Was this page helpful? Yes No

Provide product feedback

Servicing stack updates

Article • 01/22/2025 • Applies to: ✅ Windows 11, ✅ Windows 10, ✅ Windows Server

What is a servicing stack update?

Servicing stack updates provide fixes to the servicing stack, the component that installs Windows updates. Additionally, it contains the component-based servicing stack (CBS), which is a key underlying component for several elements of Windows deployment, such as:

- Deployment Image Servicing and Management (DISM)
- System File Checker (SFC)
- Changing Windows features or roles
- Component repair

CBS is a small component that typically doesn't have updates released every month.

What's the difference between a servicing stack update and a cumulative update?

Both Windows client and Windows Server use the cumulative update mechanism, in which many fixes to improve the quality and security of Windows are packaged into a single update. Each cumulative update includes the changes and fixes from all previous updates. A servicing stack update improves the reliability of the update process to mitigate potential issues while installing the latest monthly security update release and feature updates.

Starting in February 2021, the cumulative update includes the latest servicing stack updates, providing a single combined cumulative update payload for Windows Update, Windows Server Update Services (WSUS), and the Microsoft Update Catalog. This combined monthly cumulative update is available on Windows 10, version 2004 and later starting with KB4601382. If you use an endpoint management tool backed by WSUS, such as Configuration Manager, you only have to select and deploy the monthly cumulative update. The latest servicing stack updates are automatically applied correctly. Release notes and file information for cumulative updates, including notes and information related to the servicing stack, are in a single KB article.

When are they released?

Changes in the servicing stack are developed and released as part of the monthly cumulative update depending on new issues or vulnerabilities. In rare occasions, a prerequisite servicing stack update might need to be released out of band to address an issue impacting systems installing the monthly cumulative update. Out of band servicing stack updates are classified as Security with a severity rating of Critical.

Feedback

Was this page helpful? Yes No

Provide product feedback

Checkpoint cumulative updates and Microsoft Update Catalog usage

Article • 01/31/2025 •

Applies to: ✅ Windows 11, version 24H2 and later, ✅ Windows Server 2025 and later

Starting Windows 11, version 24H2, monthly security updates and optional nonsecurity preview release updates might be preceded by a checkpoint cumulative update. Devices updating from Windows Update (WU) and Windows Server Update Services (WSUS) release channels can continue to seamlessly install the latest monthly security update or the optional nonsecurity preview release regardless of whether there are any preceding checkpoint cumulative updates, so **update processes involving WU and WSUS remain unchanged**. This article covers how Microsoft Update Catalog users can easily update their devices (or images) through checkpoint cumulative updates.

Checkpoint cumulative updates

Windows 11 quality updates use servicing technology and are built cumulatively from the time when a new Windows OS was released to manufacturing (RTM). These monthly updates include all the changes since RTM in the form of binary differentials computed from the initial version of those binaries.

With Windows 11, version 24H2, Microsoft introduced a new concept of checkpoint cumulative updates. This change allows you to get features and security enhancements via the latest cumulative update through smaller, incremental differentials containing only the changes since the previous checkpoint cumulative update. This change means that you can save time, bandwidth, and hard drive space.

Going forward, Microsoft might periodically release cumulative updates as checkpoints. The subsequent updates will then consist of:

- The update package files associated with the checkpoints, and
- New update package files that contain incremental binary differentials against the version of binaries in the last checkpoint.

This process might be repeated multiple times, thereby generating multiple checkpoints during the lifecycle of a given Windows release. The Windows 11, version 24H2 servicing stack can merge all the checkpoints and only download and install content that's missing on the device.

If any checkpoint cumulative updates precede a target update, a device or image needs to take all prior checkpoint cumulative updates before it can take the target update. In other words, a post-checkpoint latest cumulative update can be applied to images/devices that are on that checkpoint or on a subsequent latest cumulative update. For updates sourced from WU and WSUS this process happens seamlessly. You can continue to use the same tools and processes that you currently use for approving and deploying updates. We expect that your experience updating through a checkpoint cumulative update will position you to efficiently take future checkpoint cumulative updates.

Applicability

A checkpoint cumulative update is just another monthly security update that informs how subsequent updates are built. There's no policy change or new requirement around when users must take these updates, though it's best practice to take monthly security updates at the earliest opportunity to keep your devices protected and productive.

Update Windows installation media

This feature doesn't introduce any change to the applicability of monthly security updates. As before, these updates apply to the main OS (install.wim) and to WinPE (boot.wim) but not to WinRE (winre.wim).

WinRE is serviced by applying the servicing stack update from a cumulative update (latest cumulative update doesn't apply) and SafeOS Dynamic Update. This is how it has been for a while now, and there's no recent change to WinRE servicing and certainly no change due to the checkpoint cumulative updates feature. We understand that not everybody may have had a shared understanding about this, but applying servicing stack update then SafeOS Dynamic Update is the only way to ensure WinRE is serviced. For more information, see Update Windows installation media with Dynamic Update.

Updating from the Microsoft Update Catalog

When installing a given monthly security or optional nonsecurity preview update, Microsoft Update Catalog users can determine and download the prior checkpoint cumulative updates and apply them sequentially under certain situations, or in one go using Deployment Image Servicing and Management (DISM).

Finding prior checkpoint cumulative updates

For a given update, users can look up the KB article and find all preceding checkpoints, if any, listed under the **Catalog** release channel. For instance, the 2024-12 monthly security update (KB5048667) has one preceding checkpoint cumulative update per December 10, 2024-KB5048667 (OS Build 26100.2605):

> **Install each MSU file individually, in order**
>
> Download and install each MSU file individually either using DISM or Windows Update Standalone Installer in the following order:
>
> - windows11.0-kb5043080-x64_953449672073f8fb99badb4cc6d5d7849b9c83e8.msu
> - windows11.0-kb5048667-x64_d4ad0ca69de9a02bc356757581e0e0d6960c9f93.msu

Alternately, users can search the KB number in the Microsoft Update Catalog and select the **Download** button for the selected architecture. The download pop-up shows all prior checkpoints for the update so that users can conveniently download all `.msu` files and apply them to their image or device. For instance, Microsoft Update Catalog shows the 2024-12 cumulative update (KB5048667) has one preceding checkpoint cumulative update, KB5043080.

Updating through checkpoint cumulative updates

Device has the latest checkpoint cumulative update and doesn't need customization:

Devices or images that have the latest checkpoint cumulative update installed and don't need Features on Demand (FoD) or language pack customization can be updated to the latest target cumulative update with no change to your existing process. You can copy the target `.msu` file from Microsoft Update Catalog and install it, for instance using Add-WindowsPackage (DISM) or DISM operating system package (.cab or .msu) servicing command-line options.

Examples of eligible devices:

Expand table

Device is on	Needs to install
• The checkpoint cumulative update, 2024-09 (KB5043080)	• A subsequent monthly security update like 2024-11 (KB5046617), or

Device is on	Needs to install
	• A subsequent optional nonsecurity release like 2024-11 (KB5046740)
• A subsequent optional nonsecurity preview release like 2024-09 (KB5043178), or • A subsequent monthly security update like 2024-10 (KB5044284)	• A subsequent monthly security update like 2025-01 (KB5050009), or • A subsequent optional nonsecurity release like 2024-11 (KB5046740)

Device needs FoD or language pack customization:

Installing FoDs or language packs requires the full latest cumulative update payload, which now can be split across files associated with each preceding checkpoint cumulative update. So, when customizing FoDs or language packs for offline media, all prior checkpoint cumulative updates and the target cumulative update need to be installed regardless of whether the device already had any of the prior checkpoints cumulative update installed. This needs to be done using DISM.

1. Copy the .msu files of the latest cumulative update (the target) and all prior checkpoint cumulative updates to a local folder. Make sure there are no other .msu files present.
2. Mount the install.wim file.
3. Run `DISM /add-package` with the latest `.msu` file as the sole target.
4. Run `/Cleanup-Image /StartComponentCleanup`.
5. Unmount.
6. Run `DISM /export-image` to optimize the image size, if that's important to you.

Device doesn't have the latest checkpoint cumulative update and doesn't need customization:

Devices that aren't on the latest checkpoint cumulative update and don't need FoD/language pack customization can either install all needed cumulative updates one by one in the right sequence. Alternately they can be updated using DISM to install all cumulative updates in one go. For more information, see the Updating through checkpoint cumulative updates section. If there are total four checkpoint cumulative updates available and device already has the first one installed, DISM applies the remaining three checkpoint cumulative updates in the right order followed by the target cumulative update, all in one go.

Related articles

- Servicing stack updates
- Features on Demand
- How to download updates that include drivers and hotfixes from the Microsoft Update Catalog
- Update Windows installation media with Dynamic Update

Feedback

Was this page helpful? Yes No

Provide product feedback

Policy CSP - Update

Update CSP policies are listed below based on the group policy area:

- Manage updates offered from Windows Update
  - AllowNonMicrosoftSignedUpdate
  - AllowOptionalContent
  - AutomaticMaintenanceWakeUp
  - BranchReadinessLevel
  - DeferFeatureUpdatesPeriodInDays
  - DeferQualityUpdatesPeriodInDays
  - DisableWUfBSafeguards
  - ExcludeWUDriversInQualityUpdate
  - ManagePreviewBuilds
  - PauseFeatureUpdates
  - PauseFeatureUpdatesStartTime
  - PauseQualityUpdates
  - PauseQualityUpdatesStartTime
  - ProductVersion
  - TargetReleaseVersion
- Manage updates offered from Windows Server Update Service
  - AllowUpdateService
  - DetectionFrequency
  - DoNotEnforceEnterpriseTLSCertPinningForUpdateDetection
  - FillEmptyContentUrls
  - SetPolicyDrivenUpdateSourceForDriverUpdates
  - SetPolicyDrivenUpdateSourceForFeatureUpdates
  - SetPolicyDrivenUpdateSourceForOtherUpdates
  - SetPolicyDrivenUpdateSourceForQualityUpdates
  - SetProxyBehaviorForUpdateDetection
  - UpdateServiceUrl
  - UpdateServiceUrlAlternate
- Manage end user experience
  - ActiveHoursEnd
  - ActiveHoursMaxRange
  - ActiveHoursStart
  - AllowAutoUpdate
  - AllowAutoWindowsUpdateDownloadOverMeteredNetwork
  - AllowMUUpdateService
  - AllowTemporaryEnterpriseFeatureControl
  - ConfigureDeadlineForFeatureUpdates

- ConfigureDeadlineForQualityUpdates
- ConfigureDeadlineGracePeriod
- ConfigureDeadlineGracePeriodForFeatureUpdates
- ConfigureDeadlineNoAutoRebootForFeatureUpdates
- ConfigureDeadlineNoAutoRebootForQualityUpdates
- ConfigureFeatureUpdateUninstallPeriod
- NoUpdateNotificationsDuringActiveHours
- ScheduledInstallDay
- ScheduledInstallEveryWeek
- ScheduledInstallFirstWeek
- ScheduledInstallFourthWeek
- ScheduledInstallSecondWeek
- ScheduledInstallThirdWeek
- ScheduledInstallTime
- SetDisablePauseUXAccess
- SetDisableUXWUAccess
- SetEDURestart
- UpdateNotificationLevel

- Legacy Policies
  - AlwaysAutoRebootAtScheduledTimeMinutes
  - AutoRestartDeadlinePeriodInDays
  - AutoRestartDeadlinePeriodInDaysForFeatureUpdates
  - AutoRestartNotificationSchedule
  - AutoRestartRequiredNotificationDismissal
  - DeferUpdatePeriod
  - DeferUpgradePeriod
  - DisableDualScan
  - EngagedRestartDeadline
  - EngagedRestartDeadlineForFeatureUpdates
  - EngagedRestartSnoozeSchedule
  - EngagedRestartSnoozeScheduleForFeatureUpdates
  - EngagedRestartTransitionSchedule
  - EngagedRestartTransitionScheduleForFeatureUpdates
  - IgnoreMOAppDownloadLimit
  - IgnoreMOUpdateDownloadLimit
  - PauseDeferrals
  - PhoneUpdateRestrictions
  - RequireDeferUpgrade
  - RequireUpdateApproval
  - ScheduleImminentRestartWarning
  - ScheduleRestartWarning

- SetAutoRestartNotificationDisable

Manage updates offered from Windows Update

AllowNonMicrosoftSignedUpdate

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1507 [10.0.10240] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/AllowNonMicrosoftSignedUpdate
```

Allows the IT admin to manage whether Automatic Updates accepts updates signed by entities other than Microsoft when the update is found at the UpdateServiceUrl location. This policy supports using WSUS for 3rd party software and patch distribution. This policy is specific to desktop and local publishing via WSUS for 3rd party updates (binaries and updates not hosted on Microsoft Update) and allows IT to manage whether Automatic Updates accepts updates signed by entities other than Microsoft when the update is found on an intranet Microsoft update service location.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	1

Allowed values:

Expand table

Value	Description
0	Not allowed or not configured. Updates from an intranet Microsoft update service location must be signed by Microsoft.
1 (Default)	Allowed. Accepts updates received through an intranet Microsoft update service location, if they're signed by a certificate found in the 'Trusted Publishers' certificate store of the local computer.

AllowOptionalContent

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 21H2 [10.0.19044.3757] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/AllowOptionalContent
```

This policy enables devices to get optional updates (including gradual feature rollouts (CFRs) - learn more by visiting aka.ms/AllowOptionalContent)

When the policy is configured.

- If "Automatically receive optional updates (including CFRs)" is selected, the device will get the latest optional updates automatically in line with the configured quality update deferrals. This includes optional cumulative updates and gradual feature rollouts (CFRs).
- If "Automatically receive optional updates" is selected, the device will only get optional cumulative updates automatically, in line with the quality update deferrals.
- If "Users can select which optional updates to receive" is selected, users can select which optional updates to get by visiting Settings > Windows Update > Advanced options > Optional updates. Users can also enable the toggle "Get the latest updates as soon as they're available" to automatically receive optional updates and gradual feature rollouts.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	Don't receive optional updates.
1	Automatically receive optional updates (including CFRs).
2	Automatically receive optional updates.
3	Users can select which optional updates to receive.

Group policy mapping:

Expand table

Name	Value
Name	AllowOptionalContent
Friendly Name	Enable optional updates
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage updates offered from Windows Update
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
Registry Value Name	SetAllowOptionalContent
ADMX File Name	WindowsUpdate.admx

AutomaticMaintenanceWakeUp

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1903 [10.0.18362] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/AutomaticMaintenanceWakeUp
```

This policy setting allows you to configure Automatic Maintenance wake up policy.

The maintenance wakeup policy specifies if Automatic Maintenance should make a wake request to the OS for the daily scheduled maintenance. Note, that if the OS power wake policy is explicitly disabled, then this setting has no effect.

- If you enable this policy setting, Automatic Maintenance will attempt to set OS wake policy and make a wake request for the daily scheduled time, if required.
- If you disable or don't configure this policy setting, the wake setting as specified in Security and Maintenance/Automatic Maintenance Control Panel will apply.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	1

Allowed values:

Expand table

Value	Description
0	Disabled.
1 (Default)	Enabled.

Group policy mapping:

Expand table

Name	Value
Name	WakeUpPolicy
Friendly Name	Automatic Maintenance WakeUp Policy
Location	Computer Configuration
Path	Windows Components > Maintenance Scheduler
Registry Key Name	Software\Policies\Microsoft\Windows\Task Scheduler\Maintenance
Registry Value Name	WakeUp
ADMX File Name	msched.admx

BranchReadinessLevel

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1607 [10.0.14393] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/BranchReadinessLevel
```

Enable this policy to specify when to receive Feature Updates.

Defer Updates | This enables devices to defer taking the next Feature Update available for their current product (or a new product if specified in the Select the target Feature Update version policy). You can defer a Feature Update for up to 14 days for all prerelease channels and up to 365 days for the General Availability Channel. To learn more about the current releases, please see aka.ms/WindowsTargetVersioninfo.

Pause Updates | To prevent Feature Updates from being offered to the device, you can temporarily pause Feature Updates. This pause will remain in effect for 35 days from the

specified start date or until the field is cleared. Note, Quality Updates will still be offered even if Feature Updates are paused.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	16

Allowed values:

Expand table

Value	Description
2	{0x2} - Windows Insider build - Fast (added in Windows 10, version 1709).
4	{0x4} - Windows Insider build - Slow (added in Windows 10, version 1709).
8	{0x8} - Release Windows Insider build (added in Windows 10, version 1709).
16 (Default)	{0x10} - Semi-Annual Enterprise Channel (Preview). Device gets all applicable feature updates from Semi-Annual Enterprise Channel (Preview).
32	2 {0x20} - Semi-Annual Enterprise Channel. Device gets feature updates from Semi-Annual Enterprise Channel. (*Only applicable to releases prior to 1903, for all releases 1903 and after the Semi-Annual Enterprise Channel and Semi-Annual Enterprise Channel (Preview) into a single Semi-Annual Enterprise Channel with a value of 16).
64	{0x40} - Release Preview of Quality Updates Only.
128	{0x80} - Canary Channel.

Group policy mapping:

Expand table

Name	Value
Name	DeferFeatureUpdates
Friendly Name	Select when Preview Builds and Feature Updates are received

Name	Value
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage updates offered from Windows Update
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

DeferFeatureUpdatesPeriodInDays

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1607 [10.0.14393] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/DeferFeatureUpdatesPeriodInDays
```

Enable this policy to specify when to receive Feature Updates.

Defer Updates | This enables devices to defer taking the next Feature Update available for their current product (or a new product if specified in the Select the target Feature Update version policy). You can defer a Feature Update for up to 14 days for all prerelease channels and up to 365 days for the General Availability Channel. To learn more about the current releases, please see aka.ms/WindowsTargetVersioninfo.

Pause Updates | To prevent Feature Updates from being offered to the device, you can temporarily pause Feature Updates. This pause will remain in effect for 35 days from the specified start date or until the field is cleared. Note, Quality Updates will still be offered even if Feature Updates are paused.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Allowed Values	Range: `[0-365]`
Default Value	0

Group policy mapping:

Expand table

Name	Value
Name	DeferFeatureUpdates
Friendly Name	Select when Preview Builds and Feature Updates are received
Element Name	How many days after a Feature Update is released would you like to defer the update before it's offered to the device?
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage updates offered from Windows Update
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

DeferQualityUpdatesPeriodInDays

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1607 [10.0.14393] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/DeferQualityUpdatesPeriodInDays
```

Enable this policy to specify when to receive quality updates.

You can defer receiving quality updates for up to 30 days.

To prevent quality updates from being received on their scheduled time, you can temporarily pause quality updates. The pause will remain in effect for 35 days or until you clear the start date field.

To resume receiving Quality Updates which are paused, clear the start date field.

If you disable or don't configure this policy, Windows Update won't alter its behavior.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Allowed Values	Range: `[0-30]`
Default Value	0

Group policy mapping:

Expand table

Name	Value
Name	DeferQualityUpdates
Friendly Name	Select when Quality Updates are received
Element Name	After a quality update is released, defer receiving it for this many days.
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage updates offered from Windows Update
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate

Name	Value
ADMX File Name	WindowsUpdate.admx

DisableWUfBSafeguards

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1809 with KB4577069 [10.0.17763.1490] and later ✅ Windows 10, version 1903 with KB4577062 [10.0.18362.1110] and later ✅ Windows 10, version 1909 with KB4577062 [10.0.18363.1110] and later ✅ Windows 10, version 2004 with KB4577063 [10.0.19041.546] and later ✅ Windows 11, version 21H2 [10.0.22000] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/DisableWUfBSafeguards
```

This policy setting specifies that a Windows Update for Business device should skip safeguards.

Safeguard holds prevent a device with a known compatibility issue from being offered a new OS version. The offering will proceed once a fix is issued and is verified on a held device. The aim of safeguards is to protect the device and user from a failed or poor upgrade experience. The safeguard holds protection is provided by default to all the devices trying to update to a new Windows 10 Feature Update version via Windows Update.

IT admins can, if necessary, opt devices out of safeguard protections using this policy setting or via the **Disable safeguards for Feature Updates** Group Policy.

> **Note**
>
> Opting out of the safeguards can put devices at risk from known performance issues. We recommend opting out only in an IT environment for validation purposes. Further, you can leverage the Windows Insider Program for Business Release Preview Channel in order to validate the upcoming Windows 10 Feature Update version without the safeguards being applied.

The disable safeguards policy will revert to "Not Configured" on a device after moving to a new Windows 10 version, even if previously enabled. This ensures the admin is consciously disabling Microsoft's default protection from known issues for each new feature update.

Disabling safeguards doesn't guarantee your device will be able to successfully update. The update may still fail on the device and will likely result in a bad experience post upgrade, as you're bypassing the protection given by Microsoft pertaining to known issues.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	Safeguards are enabled and devices may be blocked for upgrades until the safeguard is cleared.
1	Safeguards aren't enabled and upgrades will be deployed without blocking on safeguards.

ExcludeWUDriversInQualityUpdate

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1607 [10.0.14393] and later

Device
`./Device/Vendor/MSFT/Policy/Config/Update/ExcludeWUDriversInQualityUpdate`

Enable this policy to not include drivers with Windows quality updates.

If you disable or don't configure this policy, Windows Update will include updates that have a Driver classification.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	Allow Windows Update drivers.
1	Exclude Windows Update drivers.

Group policy mapping:

Expand table

Name	Value
Name	ExcludeWUDriversInQualityUpdate
Friendly Name	Do not include drivers with Windows Updates
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage updates offered from Windows Update
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate

Name	Value
Registry Value Name	ExcludeWUDriversInQualityUpdate
ADMX File Name	WindowsUpdate.admx

ManagePreviewBuilds

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1709 [10.0.16299] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/ManagePreviewBuilds
```

Used to manage Windows 10 Insider Preview builds. Value type is integer.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	3

Allowed values:

Expand table

Value	Description
0	Disable Preview builds.
1	Disable Preview builds once the next release is public.

Value	Description
2	Enable Preview builds.
3 (Default)	Preview builds is left to user selection.

Group policy mapping:

Expand table

Name	Value
Name	ManagePreviewBuilds
Friendly Name	Manage preview builds
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage updates offered from Windows Update
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

PauseFeatureUpdates

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1607 [10.0.14393] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/PauseFeatureUpdates
```

Enable this policy to specify when to receive Feature Updates.

Defer Updates | This enables devices to defer taking the next Feature Update available for their current product (or a new product if specified in the Select the target Feature Update version

policy). You can defer a Feature Update for up to 14 days for all prerelease channels and up to 365 days for the General Availability Channel. To learn more about the current releases, please see aka.ms/WindowsTargetVersioninfo.

Pause Updates | To prevent Feature Updates from being offered to the device, you can temporarily pause Feature Updates. This pause will remain in effect for 35 days from the specified start date or until the field is cleared. Note, Quality Updates will still be offered even if Feature Updates are paused.

> **Note**
>
> We recommend that you use the Update/PauseFeatureUpdatesStartTime policy, if you're running Windows 10, version 1703 or later.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	Feature Updates aren't paused.
1	Feature Updates are paused for 60 days or until value set to back to 0, whichever is sooner.

Group policy mapping:

Expand table

Name	Value
Name	DeferFeatureUpdates
Friendly Name	Select when Preview Builds and Feature Updates are received

Name	Value
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage updates offered from Windows Update
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

PauseFeatureUpdatesStartTime

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1703 [10.0.15063] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/PauseFeatureUpdatesStartTime
```

Specifies the date and time when the IT admin wants to start pausing the Feature Updates. Value type is string (yyyy-mm-dd, ex. 2018-10-28).

Description framework properties:

Expand table

Property name	Property value
Format	`chr` (string)
Access Type	Add, Delete, Get, Replace

Group policy mapping:

Expand table

Name	Value
Name	DeferFeatureUpdates
Friendly Name	Select when Preview Builds and Feature Updates are received
Element Name	Pause Preview Builds or Feature Updates starting.
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage updates offered from Windows Update
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

PauseQualityUpdates

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1607 [10.0.14393] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/PauseQualityUpdates
```

Enable this policy to specify when to receive quality updates.

You can defer receiving quality updates for up to 30 days.

To prevent quality updates from being received on their scheduled time, you can temporarily pause quality updates. The pause will remain in effect for 35 days or until you clear the start date field.

To resume receiving Quality Updates which are paused, clear the start date field.

If you disable or don't configure this policy, Windows Update won't alter its behavior.

> **Note**
>
> We recommend that you use the Update/PauseQualityUpdatesStartTime policy, if you're running Windows 10, version 1703 or later.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	Quality Updates aren't paused.
1	Quality Updates are paused for 35 days or until value set back to 0, whichever is sooner.

Group policy mapping:

Expand table

Name	Value
Name	DeferQualityUpdates
Friendly Name	Select when Quality Updates are received
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage updates offered from Windows Update
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

PauseQualityUpdatesStartTime

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1703 [10.0.15063] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/PauseQualityUpdatesStartTime
```

Specifies the date and time when the IT admin wants to start pausing the Quality Updates. Value type is string (yyyy-mm-dd, ex. 2018-10-28).

> **Note**
>
> When this policy is configured, Quality Updates will be paused for 35 days from the specified start date.

Description framework properties:

Expand table

Property name	Property value
Format	`chr` (string)
Access Type	Add, Delete, Get, Replace

Group policy mapping:

Expand table

Name	Value
Name	DeferQualityUpdates
Friendly Name	Select when Quality Updates are received
Element Name	Pause Quality Updates starting.

Name	Value
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage updates offered from Windows Update
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

ProductVersion

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 2004 with KB5005101 [10.0.19041.1202] and later ✅ Windows 10, version 20H2 with KB5005101 [10.0.19042.1202] and later ✅ Windows 10, version 21H1 with KB5005101 [10.0.19043.1202] and later ✅ Windows 11, version 21H2 [10.0.22000] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/ProductVersion
```

Enables IT administrators to specify the product version associated with the target feature update they would like their device(s) to move to and/or stay on until they reach end of service or reconfigure the policy. For details about different Windows 10 versions, see Windows release information.

Supported value type is a string containing a Windows product. For example, "Windows 11" or "11" or "Windows 10". By using this Windows Update for Business policy to upgrade devices to a new product (for example, Windows 11) you're agreeing that when applying this operating system to a device:

1. The applicable Windows license was purchased through volume licensing, or
2. You're authorized to bind your organization and are accepting on its behalf the relevant Microsoft Software License Terms.

> **Note**
>
> If no product is specified, the device will continue receiving newer versions of the Windows product it's currently on.

Description framework properties:

Expand table

Property name	Property value
Format	`chr` (string)
Access Type	Add, Delete, Get, Replace

Group policy mapping:

Expand table

Name	Value
Name	TargetReleaseVersion
Friendly Name	Select the target Feature Update version
Element Name	Which Windows product version would you like to receive feature updates for? e.g., Windows 10.
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage updates offered from Windows Update
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

TargetReleaseVersion

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1803 with KB4556807 [10.0.17134.1488] and later ✅ Windows 10, version 1809 with KB4551853 [10.0.17763.1217] and later ✅ Windows 10, version 1903 with KB4556799 [10.0.18362.836] and later ✅ Windows 10, version 1909 with KB4556799 [10.0.18363.836] and later ✅ Windows 10, version 2004 [10.0.19041] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/TargetReleaseVersion
```

Enables IT administrators to specify which version they would like their device(s) to move to and/or stay on until they reach end of service or reconfigure the policy. For details about different Windows 10 versions, see Windows 10 release information.

Supported value type is a string containing Windows version number. For example, `1809`, `1903`, etc.

> **Note**
>
> You need to set up the ProductVersion CSP along with the TargetReleaseVersion CSP for it to work.

Description framework properties:

Expand table

Property name	Property value
Format	`chr` (string)
Access Type	Add, Delete, Get, Replace

Group policy mapping:

Expand table

Name	Value
Name	TargetReleaseVersion

Name	Value
Friendly Name	Select the target Feature Update version
Element Name	Target Version for Feature Updates.
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage updates offered from Windows Update
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

Manage updates offered from Windows Server Update Service

AllowUpdateService

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1507 [10.0.10240] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/AllowUpdateService
```

Specifies whether the device could use Microsoft Update, Windows Server Update Services (WSUS), or Microsoft Store. Even when Windows Update is configured to receive updates from an intranet update service, it will periodically retrieve information from the public Windows Update service to enable future connections to Windows Update, and other services like Microsoft Update or the Microsoft Store. Enabling this policy will disable that functionality, and may cause connection to public services such as the Microsoft Store to stop working.

ⓘ Note

This policy applies only when the desktop or device is configured to connect to an intranet update service using the Specify intranet Microsoft update service location policy.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	1

Allowed values:

Expand table

Value	Description
0	Not allowed.
1 (Default)	Allowed.

Group policy mapping:

Expand table

Name	Value
Name	CorpWuURL
Friendly Name	Specify intranet Microsoft update service location
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage updates offered from Windows Server Update Service
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

DetectionFrequency

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1703 [10.0.15063] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/DetectionFrequency
```

Specifies the hours that Windows will use to determine how long to wait before checking for available updates. The exact wait time is a sum of the specific value and a random variant of 0-4 hours.

If the status is set to Enabled, Windows will check for available updates at the specified interval.

If the status is set to Disabled or Not Configured, Windows will check for available updates at the default interval of 22 hours.

> **Note**
>
> The "Specify intranet Microsoft update service location" setting must be enabled for this policy to have effect.

> **Note**
>
> If the "Configure Automatic Updates" policy is disabled, this policy has no effect.

> **Note**
>
> This policy isn't supported on Windows RT. Setting this policy won't have any effect on Windows RT PCs.

This policy should be enabled only when UpdateServiceUrl is configured to point the device at a WSUS server rather than Microsoft Update.

> **Note**

There is a random variant of 0-4 hours applied to the scan frequency, which cannot be configured.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Allowed Values	Range: `[1-22]`
Default Value	22

Group policy mapping:

Expand table

Name	Value
Name	DetectionFrequency_Title
Friendly Name	Automatic Updates detection frequency
Element Name	interval (hours)
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage updates offered from Windows Server Update Service
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate\AU
ADMX File Name	WindowsUpdate.admx

DoNotEnforceEnterpriseTLSCertPinningForUpdateDetection

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1507 with KB4598231 [10.0.10240.18818] and later ✅ Windows 10, version 1607 with KB4598243 [10.0.14393.4169] and later ✅ Windows 10, version 1703 with KB4520010 [10.0.15063.2108] and later ✅ Windows 10, version 1709 with KB4580328 [10.0.16299.2166] and later ✅ Windows 10, version 1803 with KB4598245 [10.0.17134.1967] and later ✅ Windows 10, version 1809 with KB4598230 [10.0.17763.1697] and later ✅ Windows 10, version 1903 [10.0.18362.1316] and later ✅ Windows 10, version 1909 with KB4598229 [10.0.18363.1316] and later ✅ Windows 10, version 2004 with KB4598242 [10.0.19041.746] and later ✅ Windows 10, version 20H2 with KB4598242 [10.0.19042.746] and later ✅ Windows 11, version 21H2 [10.0.22000] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/DoNotEnforceEnterpriseTLSCertPinningForUpdateDetection
```

> **Note**
>
> By default, certificate pinning for Windows Update client isn't enforced. To ensure the highest levels of security, we recommended using WSUS TLS certificate pinning on all devices.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	Disabled.
1	Enabled.

Group policy mapping:

Expand table

Name	Value
Name	CorpWuURL
Friendly Name	Specify intranet Microsoft update service location
Element Name	Don't enforce TLS certificate pinning for Windows Update client for detecting updates.
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage updates offered from Windows Server Update Service
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

FillEmptyContentUrls

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1607 [10.0.14393] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/FillEmptyContentUrls
```

Allows Windows Update Agent to determine the download URL when it's missing from the metadata. This scenario will occur when intranet update service stores the metadata files but the download contents are stored in the ISV file cache (specified as the alternate download URL).

> **Note**
>
> This setting should only be used in combination with an alternate download URL and configured to use ISV file cache. This setting is used when the intranet update service doesn't provide download URLs in the update metadata for files which are available on the alternate download server.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	Disabled.
1	Enabled.

Group policy mapping:

Expand table

Name	Value
Name	CorpWuURL
Friendly Name	Specify intranet Microsoft update service location

Name	Value
Element Name	Download files with no Url in the metadata if alternate download server is set.
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage updates offered from Windows Server Update Service
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

SetPolicyDrivenUpdateSourceForDriverUpdates

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ [10.0.20348.371] and later ✅ Windows 10, version 2004 with KB5005101 [10.0.19041.1202] and later ✅ Windows 10, version 20H2 with KB5005101 [10.0.19042.1202] and later ✅ Windows 10, version 21H1 with KB5005101 [10.0.19043.1202] and later ✅ Windows 10, version 21H2 [10.0.19044.1288] and later ✅ Windows 10, version 22H2 [10.0.19045.2130] and later ✅ Windows 11, version 21H2 [10.0.22000] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/SetPolicyDrivenUpdateSourceForDriverUpdates
```

Configure this policy to specify whether to receive **Windows Driver Updates** from Windows Update endpoint, managed by Windows Update for Business policies, or through your configured Windows Server Update Service (WSUS) server. If you configure this policy, also configure the scan source policies for other update types:

- SetPolicyDrivenUpdateSourceForFeatureUpdates
- SetPolicyDrivenUpdateSourceForQualityUpdates
- SetPolicyDrivenUpdateSourceForOtherUpdates

> **Note**
>
> If you haven't properly configured Update/UpdateServiceUrl correctly to point to your WSUS server, this policy has no effect.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	1

Allowed values:

Expand table

Value	Description
0	Detect, download and deploy Driver Updates from Windows Update.
1 (Default)	Detect, download and deploy Driver Updates from Windows Server Update Services (WSUS).

Group policy mapping:

Expand table

Name	Value
Name	CorpWuURL
Friendly Name	Specify intranet Microsoft update service location
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage updates offered from Windows Server Update Service
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

SetPolicyDrivenUpdateSourceForFeatureUpdates

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ [10.0.20348.371] and later ✅ Windows 10, version 2004 with KB5005101 [10.0.19041.1202] and later ✅ Windows 10, version 20H2 with KB5005101 [10.0.19042.1202] and later ✅ Windows 10, version 21H1 with KB5005101 [10.0.19043.1202] and later ✅ Windows 10, version 21H2 [10.0.19044.1288] and later ✅ Windows 10, version 22H2 [10.0.19045.2130] and later ✅ Windows 11, version 21H2 [10.0.22000] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/SetPolicyDrivenUpdateSourceForFeatureUpdates
```

Configure this policy to specify whether to receive **Windows Feature Updates** from Windows Update endpoint, managed by Windows Update for Business policies, or through your configured Windows Server Update Service (WSUS) server. If you configure this policy, also configure the scan source policies for other update types:

- SetPolicyDrivenUpdateSourceForQualityUpdates
- SetPolicyDrivenUpdateSourceForDriverUpdates
- SetPolicyDrivenUpdateSourceForOtherUpdates

> **Note**
>
> - If you haven't properly configured Update/UpdateServiceUrl correctly to point to your WSUS server, this policy has no effect.
> - If you're also using the **Specify settings for optional component installation and component repair** (ADMX_Servicing) policy to enable content for FoDs and language packs, see How to make Features on Demand and language packs available when you're using WSUS or Configuration Manager to verify your policy configuration.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	1

Allowed values:

Expand table

Value	Description
0	Detect, download and deploy Feature Updates from Windows Update.
1 (Default)	Detect, download and deploy Feature Updates from Windows Server Update Services (WSUS).

Group policy mapping:

Expand table

Name	Value
Name	CorpWuURL
Friendly Name	Specify intranet Microsoft update service location
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage updates offered from Windows Server Update Service
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

SetPolicyDrivenUpdateSourceForOtherUpdates

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ [10.0.20348.371] and later ✅ Windows 10, version 2004 with KB5005101 [10.0.19041.1202] and later ✅ Windows 10, version 20H2 with KB5005101 [10.0.19042.1202] and later ✅ Windows 10, version 21H1 with KB5005101 [10.0.19043.1202] and later ✅ Windows 10, version 21H2 [10.0.19044.1288] and later ✅ Windows 10, version 22H2 [10.0.19045.2130] and later ✅ Windows 11, version 21H2 [10.0.22000] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/SetPolicyDrivenUpdateSourceForOtherUpdates
```

Configure this policy to specify whether to receive **Other Updates** from Windows Update endpoint, managed by Windows Update for Business policies, or through your configured Windows Server Update Service (WSUS) server. If you configure this policy, also configure the scan source policies for other update types:

- SetPolicyDrivenUpdateSourceForFeatureUpdates
- SetPolicyDrivenUpdateSourceForQualityUpdates
- SetPolicyDrivenUpdateSourceForDriverUpdates

> **Note**
>
> If you haven't properly configured Update/UpdateServiceUrl correctly to point to your WSUS server, this policy has no effect.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	1

Allowed values:

Expand table

Value	Description
0	Detect, download and deploy other Updates from Windows Update.
1 (Default)	Detect, download and deploy other Updates from Windows Server Update Services (WSUS).

Group policy mapping:

Expand table

Name	Value
Name	CorpWuURL
Friendly Name	Specify intranet Microsoft update service location
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage updates offered from Windows Server Update Service
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

SetPolicyDrivenUpdateSourceForQualityUpdates

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ [10.0.20348.371] and later ✅ Windows 10, version 2004 with KB5005101 [10.0.19041.1202] and later ✅ Windows 10, version 20H2 with KB5005101 [10.0.19042.1202] and later ✅ Windows 10, version 21H1 with KB5005101 [10.0.19043.1202] and later ✅ Windows 10, version 21H2 [10.0.19044.1288] and later ✅ Windows 10, version 22H2 [10.0.19045.2130] and later ✅ Windows 11, version 21H2 [10.0.22000] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/SetPolicyDrivenUpdateSourceForQualityUpdates
```

Configure this policy to specify whether to receive **Windows Quality Updates** from Windows Update endpoint, managed by Windows Update for Business policies, or through your configured Windows Server Update Service (WSUS) server. If you configure this policy, also configure the scan source policies for other update types:

- SetPolicyDrivenUpdateSourceForFeatureUpdates
- SetPolicyDrivenUpdateSourceForDriverUpdates
- SetPolicyDrivenUpdateSourceForOtherUpdates

> **Note**
>
> - If you haven't properly configured Update/UpdateServiceUrl correctly to point to your WSUS server, this policy has no effect.
> - If you're also using the **Specify settings for optional component installation and component repair** (ADMX_Servicing) policy to enable content for FoDs and language packs, see How to make Features on Demand and language packs available when you're using WSUS or Configuration Manager to verify your policy configuration.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	1

Allowed values:

Expand table

Value	Description
0	Detect, download and deploy Quality Updates from Windows Update.

Value	Description
1 (Default)	Detect, download and deploy Quality Updates from Windows Server Update Services (WSUS).

Group policy mapping:

Expand table

Name	Value
Name	CorpWuURL
Friendly Name	Specify intranet Microsoft update service location
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage updates offered from Windows Server Update Service
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

SetProxyBehaviorForUpdateDetection

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1507 with KB4577049 [10.0.10240.18696] and later ✅ Windows 10, version 1607 with KB4577015 [10.0.14393.3930] and later ✅ Windows 10, version 1703 [10.0.15063.2500] and later ✅ Windows 10, version 1709 with KB4577041 [10.0.16299.2107] and later ✅ Windows 10, version 1803 with KB4577032 [10.0.17134.1726] and later ✅ Windows 10, version 1809 with KB4570333 [10.0.17763.1457] and later ✅ Windows 10, version 1903 with KB4574727 [10.0.18362.1082] and later ✅ Windows 10, version 1909 with KB4574727 [10.0.18363.1082] and later ✅ Windows 10, version 2004 with KB4571756

Scope	Editions	Applicable OS
		[10.0.19041.508] and later ✅ Windows 11, version 21H2 [10.0.22000] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/SetProxyBehaviorForUpdateDetection
```

By default, HTTP WSUS servers scan only if system proxy is configured. This policy setting allows you to configure user proxy as a fallback for detecting updates while using an HTTP-based intranet server despite the vulnerabilities it presents.

This policy setting doesn't affect those customers who have, per Microsoft recommendation, secured their WSUS server with TLS/SSL protocol, thereby using HTTPS-based intranet servers to keep systems secure. That said, if a proxy is required, we recommend configuring a system proxy to ensure the highest level of security.

> **Note**
>
> Configuring this policy setting to 1 exposes your environment to potential security risk and makes scans unsecure.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	Only use system proxy for detecting updates (default).
1	Allow user proxy to be used as a fallback if detection using system proxy fails.

Group policy mapping:

Expand table

Name	Value
Name	CorpWuURL
Friendly Name	Specify intranet Microsoft update service location
Element Name	Select the proxy behavior for Windows Update client for detecting updates.
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage updates offered from Windows Server Update Service
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

UpdateServiceUrl

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1507 [10.0.10240] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/UpdateServiceUrl
```

Allows the device to check for updates from a WSUS server instead of Microsoft Update. This is useful for on-premises MDMs that need to update devices that can't connect to the Internet.

The following list shows the supported values:

- Not configured: The device checks for updates from Microsoft Update.
- Set to a URL, such as `http://abcd-srv:8530`: The device checks for updates from the WSUS server at the specified URL.

Description framework properties:

Expand table

Property name	Property value
Format	`chr` (string)
Access Type	Add, Delete, Get, Replace
Default Value	CorpWSUS

Group policy mapping:

Expand table

Name	Value
Name	CorpWuURL
Friendly Name	Specify intranet Microsoft update service location
Element Name	Set the intranet update service for detecting updates.
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage updates offered from Windows Server Update Service
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

Example:

XML

```xml
<Replace>
    <CmdID>$CmdID$</CmdID>
    <Item>
        <Meta>
            <Format>chr</Format>
            <Type>text/plain</Type>
        </Meta>
        <Target>
            <LocURI>./Vendor/MSFT/Policy/Config/Update/UpdateServiceUrl</LocURI>
        </Target>
        <Data>http://abcd-srv:8530</Data>
```

```
    </Item>
</Replace>
```

UpdateServiceUrlAlternate

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1607 [10.0.14393] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/UpdateServiceUrlAlternate
```

Specifies an alternate intranet server to host updates from Microsoft Update. You can then use this update service to automatically update computers on your network. This setting lets you specify a server on your network to function as an internal update service. The Automatic Updates client will search this service for updates that apply to the computers on your network. To use this setting, you must set two server name values: the server from which the Automatic Updates client detects and downloads updates, and the server to which updated workstations upload statistics. You can set both values to be the same server. An optional server name value can be specified to configure Windows Update agent, and download updates from an alternate download server instead of WSUS Server. Value type is string and the default value is an empty string, . If the setting isn't configured, and if Automatic Updates isn't disabled by policy or user preference, the Automatic Updates client connects directly to the Windows Update site on the Internet.

> **Note**
>
> If the Configure Automatic Updates Group Policy is disabled, then this policy has no effect. If the Alternate Download Server Group Policy isn't set, it will use the WSUS server by default to download updates. This policy isn't supported on Windows RT. Setting this policy won't have any effect on Windows RT PCs.

Description framework properties:

Expand table

Property name	Property value
Format	`chr` (string)
Access Type	Add, Delete, Get, Replace

Group policy mapping:

Expand table

Name	Value
Name	CorpWuURL
Friendly Name	Specify intranet Microsoft update service location
Element Name	Set the alternate download server.
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage updates offered from Windows Server Update Service
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

Manage end user experience

ActiveHoursEnd

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1607 [10.0.14393] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/ActiveHoursEnd
```

- If you enable this policy, the PC won't automatically restart after updates during active hours. The PC will attempt to restart outside of active hours.

Note that the PC must restart for certain updates to take effect.

- If you disable or don't configure this policy and have no other reboot group policies, the user selected active hours will be in effect.

If any of the following two policies are enabled, this policy has no effect:

1. No auto-restart with logged-on users for scheduled automatic updates installations.
2. Always automatically restart at scheduled time.

Note that the default max active hours range is 18 hours from the active hours start time unless otherwise configured via the Specify active hours range for auto-restarts policy.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Allowed Values	Range: `[0-23]`
Default Value	17

Group policy mapping:

Expand table

Name	Value
Name	ActiveHours
Friendly Name	Turn off auto-restart for updates during active hours
Element Name	End.
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage end user experience

Name	Value
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

ActiveHoursMaxRange

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1703 [10.0.15063] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/ActiveHoursMaxRange
```

Enable this policy to specify the maximum number of hours from the start time that users can set their active hours.

The max active hours range can be set between 8 and 18 hours.

If you disable or don't configure this policy, the default max active hours range will be used.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Allowed Values	Range: `[8-18]`
Default Value	18

Group policy mapping:

Expand table

Name	Value
Name	ActiveHoursMaxRange
Friendly Name	Specify active hours range for auto-restarts
Element Name	Max range.
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage end user experience
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

ActiveHoursStart

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1607 [10.0.14393] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/ActiveHoursStart
```

- If you enable this policy, the PC won't automatically restart after updates during active hours. The PC will attempt to restart outside of active hours.

Note that the PC must restart for certain updates to take effect.

- If you disable or don't configure this policy and have no other reboot group policies, the user selected active hours will be in effect.

If any of the following two policies are enabled, this policy has no effect:

1. No auto-restart with logged-on users for scheduled automatic updates installations.
2. Always automatically restart at scheduled time.

Note that the default max active hours range is 18 hours from the active hours start time unless otherwise configured via the Specify active hours range for auto-restarts policy.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Allowed Values	Range: `[0-23]`
Default Value	8

Group policy mapping:

Expand table

Name	Value
Name	ActiveHours
Friendly Name	Turn off auto-restart for updates during active hours
Element Name	Start.
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage end user experience
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

AllowAutoUpdate

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1507 [10.0.10240] and later

Device
`./Device/Vendor/MSFT/Policy/Config/Update/AllowAutoUpdate`

Enables the IT admin to manage automatic update behavior to scan, download, and install updates. Important. This option should be used only for systems under regulatory compliance, as you won't get security updates as well. If the policy isn't configured, end-users get the default behavior (Auto install and restart).

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	2

Allowed values:

Expand table

Value	Description
0	Notify the user before downloading the update. This policy is used by the enterprise who wants to enable the end-users to manage data usage. With this option users are notified when there are updates that apply to the device and are ready for download. Users can download and install the updates from the Windows Update control panel.
1	Auto install the update and then notify the user to schedule a device restart. Updates are downloaded automatically on non-metered networks and installed during "Automatic Maintenance" when the device isn't in use and isn't running on battery power. If automatic maintenance is unable to install updates for two days, Windows Update will install updates immediately. If the installation requires a restart, the end-user is prompted to schedule the restart time. After the update is installed, if the user hasn't scheduled a restart, the device will attempt to restart automatically. The user will be notified about the scheduled restart and can reschedule it if the proposed time is inconvenient. Enabling the end-user to control the start time reduces the risk of accidental data loss caused by applications that don't shutdown properly on restart.
2 (Default)	Auto install and restart. Updates are downloaded automatically on non-metered networks and installed during "Automatic Maintenance" when the device isn't in use and isn't running on battery power. If automatic maintenance is unable to install updates for two days, Windows Update will install updates right away. If a restart is required, then the device is automatically restarted when the device isn't actively being used. This is the default behavior for unmanaged

Value	Description
	devices. Devices are updated quickly, but it increases the risk of accidental data loss caused by an application that doesn't shutdown properly on restart.
3	Auto install and restart at a specified time. The IT specifies the installation day and time. If no day and time are specified, the default is 3 AM daily. Automatic installation happens at this time and device restart happens after a 15-minute countdown. If the user is logged in when Windows is ready to restart, the user can interrupt the 15-minute countdown to delay the restart.
4	Auto install and restart without end-user control. Updates are downloaded automatically on non-metered networks and installed during "Automatic Maintenance" when the device isn't in use and isn't running on battery power. If automatic maintenance is unable to install updates for two days, Windows Update will install updates right away. If a restart is required, then the device is automatically restarted when the device isn't actively being used. This setting option also sets the end-user control panel to read-only.
5	Turn off automatic updates.

Group policy mapping:

Expand table

Name	Value
Name	AutoUpdateCfg
Friendly Name	Configure Automatic Updates
Element Name	Configure automatic updating.
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage end user experience
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate\AU
ADMX File Name	WindowsUpdate.admx

AllowAutoWindowsUpdateDownloadOverMeteredNetwork

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education	✅ Windows 10, version 1709 [10.0.16299] and later

Scope	Editions	Applicable OS
	✅ IoT Enterprise / IoT Enterprise LTSC	

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/AllowAutoWindowsUpdateDownloadOverMeteredNetwork
```

Enabling this policy will automatically download updates, even over metered data connections (charges may apply)

A significant number of devices primarily use cellular data and don't have Wi-Fi access, which leads to a lower number of devices getting updates. Since a large number of devices have large data plans or unlimited data, this policy can unblock devices from getting updates.

This policy is accessible through the Update setting in the user interface or Group Policy.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	Not allowed.
1	Allowed.

Group policy mapping:

Expand table

Name	Value
Name	AllowAutoWindowsUpdateDownloadOverMeteredNetwork
Friendly Name	Allow updates to be downloaded automatically over metered connections
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage end user experience
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
Registry Value Name	AllowAutoWindowsUpdateDownloadOverMeteredNetwork
ADMX File Name	WindowsUpdate.admx

AllowMUUpdateService

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1607 [10.0.14393] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/AllowMUUpdateService
```

Allows the IT admin to manage whether to scan for app updates from Microsoft Update.

> **Note**
>
> - For a list of other Microsoft products that might be updated, see Update other Microsoft products.
> - Setting this policy back to 0 or Not configured doesn't revert the configuration to receive updates from Microsoft Update automatically. In order to revert the configuration, you can run the PowerShell commands that are listed below to remove the Microsoft Update service:
>
> PowerShell

```
$MUSM = New-Object -ComObject "Microsoft.Update.ServiceManager"
$MUSM.RemoveService("7971f918-a847-4430-9279-4a52d1efe18d")
```

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	Not allowed or not configured.
1	Allowed. Accepts updates received through Microsoft Update.

Group policy mapping:

Expand table

Name	Value
Name	AutoUpdateCfg
Friendly Name	Configure Automatic Updates
Element Name	Install updates for other Microsoft products.
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage end user experience
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate\AU
ADMX File Name	WindowsUpdate.admx

AllowTemporaryEnterpriseFeatureControl

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 11, version 22H2 with KB5022913 [10.0.22621.1344] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/AllowTemporaryEnterpriseFeatureControl
```

Features introduced via servicing (outside of the annual feature update) are off by default for devices that have their Windows updates managed*.

- If this policy is configured to "Enabled", then all features available in the latest monthly quality update installed will be on.
- If this policy is set to "Not Configured" or "Disabled" then features that are shipped via a monthly quality update (servicing) will remain off until the feature update that includes these features is installed.

*Windows update managed devices are those that have their Windows updates managed via policy; whether via the cloud using Windows Update for Business or on-premises with Windows Server Update Services (WSUS).

> **Note**
>
> In Intune, this setting is known as **Allow Temporary Enterprise Feature Control** and is available in the Settings Catalog. By default, all features introduced via servicing that are behind the commercial control are off for Windows-Update-managed devices. When set to Allowed, these features are enabled and turned on. For more information, see Blog: Commercial control for continuous innovation.

Description framework properties:

Expand table

Property name	Property value
Format	`int`

Property name	Property value
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	Not allowed.
1	Allowed.

Group policy mapping:

Expand table

Name	Value
Name	AllowTemporaryEnterpriseFeatureControl
Friendly Name	Enable features introduced via servicing that are off by default
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage end user experience
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
Registry Value Name	AllowTemporaryEnterpriseFeatureControl
ADMX File Name	WindowsUpdate.admx

ConfigureDeadlineForFeatureUpdates

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1903 [10.0.18362] and later

Device
`./Device/Vendor/MSFT/Policy/Config/Update/ConfigureDeadlineForFeatureUpdates`

Number of days before feature updates are installed on devices automatically regardless of active hours. Before the deadline passes, users can schedule restarts, and automatic restarts can happen outside of active hours. When set to 0, updates will download and install immediately, but might not finish within the day due to device availability and network connectivity.

> **Note**
>
> - After the deadline passes, restarts will occur regardless of active hours and users won't be able to reschedule.
> - When this policy is used, the download, installation, and reboot settings from Update/AllowAutoUpdate are ignored.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Allowed Values	Range: `[0-30]`
Default Value	2

Group policy mapping:

Expand table

Name	Value
Name	ComplianceDeadlineForFU
Friendly Name	Specify deadline for automatic updates and restarts for feature update
Element Name	Deadline (days)
Location	Computer Configuration

Name	Value
Path	Windows Components > Windows Update > Manage end user experience
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

ConfigureDeadlineForQualityUpdates

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1903 [10.0.18362] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/ConfigureDeadlineForQualityUpdates
```

Number of days before quality updates are installed on devices automatically regardless of active hours. Before the deadline passes, users can schedule restarts, and automatic restarts can happen outside of active hours. When set to 0, updates will download and install immediately, but might not finish within the day due to device availability and network connectivity.

> **Note**
>
> - After the deadline passes, restarts will occur regardless of active hours and users won't be able to reschedule.
> - When this policy is used, the download, installation, and reboot settings from Update/AllowAutoUpdate are ignored.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Allowed Values	Range: `[0-30]`
Default Value	7

Group policy mapping:

Expand table

Name	Value
Name	ComplianceDeadline
Friendly Name	Specify deadline for automatic updates and restarts for quality update
Element Name	Deadline (days)
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage end user experience
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

ConfigureDeadlineGracePeriod

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1903 [10.0.18362] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/ConfigureDeadlineGracePeriod
```

Minimum number of days from update installation until restarts occur automatically for quality updates. This policy only takes effect when Update/ConfigureDeadlineForQualityUpdates is configured. If Update/ConfigureDeadlineForQualityUpdates is configured but this policy isn't, then the default value of 2 days will take effect.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Allowed Values	Range: `[0-7]`
Default Value	2

Group policy mapping:

Expand table

Name	Value
Name	ComplianceDeadline
Friendly Name	Specify deadline for automatic updates and restarts for quality update
Element Name	Grace period (days)
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage end user experience
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

ConfigureDeadlineGracePeriodForFeatureUpdates

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1809 with KB5000854 [10.0.17763.1852] and later ✅ Windows 10, version 1909 with KB5000850 [10.0.18363.1474] and later ✅ Windows 10, version 2004 with KB5000842 [10.0.19041.906] and later ✅ Windows 10, version 20H2 with KB5000842 [10.0.19042.906] and later ✅ Windows 11, version 21H2 [10.0.22000] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/ConfigureDeadlineGracePeriodForFeatureUpdates
```

Minimum number of days from update installation until restarts occur automatically for feature updates. This policy only takes effect when Update/ConfigureDeadlineForFeatureUpdates is configured. If Update/ConfigureDeadlineForFeatureUpdates is configured but this policy isn't, then the value configured by Update/ConfigureDeadlineGracePeriod will be used. If Update/ConfigureDeadlineGracePeriod is also not configured, then the default value of 7 days will take effect.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Allowed Values	Range: `[0-7]`
Default Value	7

Group policy mapping:

Expand table

Name	Value
Name	ComplianceDeadlineForFU
Friendly Name	Specify deadline for automatic updates and restarts for feature update

Name	Value
Element Name	Grace Period (days)
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage end user experience
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

ConfigureDeadlineNoAutoRebootForFeatureUpdates

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 11, version 22H2 [10.0.22621] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/ConfigureDeadlineNoAutoRebootForFeatureUpdates
```

This policy lets you specify the number of days before feature updates are installed on devices automatically, and a grace period after which required restarts occur automatically.

Set deadlines for feature updates and quality updates to meet your compliance goals. Updates will be downloaded and installed as soon as they're offered and automatic restarts will be attempted outside of active hours. Once the deadline has passed, restarts will occur regardless of active hours, and users won't be able to reschedule. If the deadline is set to 0 days, the update will be installed immediately upon offering, but might not finish within the day due to device availability and network connectivity.

Set a grace period for feature updates to guarantee users a minimum time to manage their restarts once updates are installed. Users can schedule restarts during the grace period and Windows can still automatically restart outside of active hours if users choose not to schedule restarts. The grace period might not take effect if users already have more than the number of days set as grace period to manage their restart, based on deadline configurations.

You can set the device to delay restarting until both the deadline and grace period have expired.

If you disable or don't configure this policy, devices will get updates and will restart according to the default schedule.

This policy overrides the following policies:

1. Specify deadline before auto restart for update installation
2. Specify Engaged restart transition and notification schedule for updates.
3. Always automatically restart at the scheduled time
4. Configure Automatic Updates.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	Disabled.
1	Enabled.

Group policy mapping:

Expand table

Name	Value
Name	ComplianceDeadlineForFU
Friendly Name	Specify deadline for automatic updates and restarts for feature update

Name	Value
Element Name	Don't auto-restart until end of grace period.
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage end user experience
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

ConfigureDeadlineNoAutoRebootForQualityUpdates

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 11, version 22H2 [10.0.22621] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/ConfigureDeadlineNoAutoRebootForQualityUpdates
```

This policy lets you specify the number of days before quality updates are installed on devices automatically, and a grace period after which required restarts occur automatically.

Set deadlines for quality updates to meet your compliance goals. Updates will be downloaded and installed as soon as they're offered and automatic restarts will be attempted outside of active hours. Once the deadline has passed, restarts will occur regardless of active hours, and users won't be able to reschedule. If the deadline is set to 0 days, the update will be installed immediately upon offering, but might not finish within the day due to device availability and network connectivity.

Set a grace period for quality updates to guarantee users a minimum time to manage their restarts once updates are installed. Users can schedule restarts during the grace period and Windows can still automatically restart outside of active hours if users choose not to schedule restarts. The grace period might not take effect if users already have more than the number of days set as grace period to manage their restart, based on deadline configurations.

You can set the device to delay restarting until both the deadline and grace period have expired.

If you disable or don't configure this policy, devices will get updates and will restart according to the default schedule.

This policy overrides the following policies:

1. Specify deadline before auto restart for update installation
2. Specify Engaged restart transition and notification schedule for updates.
3. Always automatically restart at the scheduled time
4. Configure Automatic Updates.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	Disabled.
1	Enabled.

Group policy mapping:

Expand table

Name	Value
Name	ComplianceDeadline
Friendly Name	Specify deadline for automatic updates and restarts for quality update

Name	Value
Element Name	Don't auto-restart until end of grace period.
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage end user experience
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

ConfigureFeatureUpdateUninstallPeriod

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1803 [10.0.17134] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/ConfigureFeatureUpdateUninstallPeriod
```

Enable enterprises/IT admin to configure feature update uninstall period.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Allowed Values	Range: `[2-60]`
Default Value	10

NoUpdateNotificationsDuringActiveHours

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 11, version 22H2 [10.0.22621] and later

Device
`./Device/Vendor/MSFT/Policy/Config/Update/NoUpdateNotificationsDuringActiveHours`

0 (default) - Use the default Windows Update notifications 1 - Turn off all notifications, excluding restart warnings 2 - Turn off all notifications, including restart warnings.

This policy allows you to define what Windows Update notifications users see. This policy doesn't control how and when updates are downloaded and installed.

Important if you choose not to get update notifications and also define other Group policy so that devices aren't automatically getting updates, neither you nor device users will be aware of critical security, quality, or feature updates, and your devices may be at risk.

If you select "Apply only during active hours" in conjunction with Option 1 or 2, then notifications will only be disabled during active hours. You can set active hours by setting "Turn off auto-restart for updates during active hours" or allow the device to set active hours based on user behavior. To ensure that the device stays secure, a notification will still be shown if this option is selected once "Specify deadlines for automatic updates and restarts" deadline has been reached if configured, regardless of active hours.

> **Note**
>
> This policy can be used in conjunction with Update/ActiveHoursStart and Update/ActiveHoursEnd policies to ensure that the end user sees no update notifications during active hours until deadline is reached. If no active hour period is configured then this will apply to the intelligent active hours window calculated on the device.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	Disabled.
1	Enabled.

Group policy mapping:

Expand table

Name	Value
Name	UpdateNotificationLevel
Friendly Name	Display options for update notifications
Element Name	Apply only during active hours.
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage end user experience
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

ScheduledInstallDay

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education	✅ Windows 10, version 1507 [10.0.10240] and later

Scope	Editions	Applicable OS
	✅ IoT Enterprise / IoT Enterprise LTSC	

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/ScheduledInstallDay
```

Enables the IT admin to schedule the day of the update installation. The data type is a integer.

> **Note**
>
> This policy will only take effect if Update/AllowAutoUpdate has been configured to option 3 or 4 for scheduled installation.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	Every day.
1	Sunday.
2	Monday.
3	Tuesday.
4	Wednesday.
5	Thursday.
6	Friday.

Value	Description
7	Saturday.

Group policy mapping:

Expand table

Name	Value
Name	AutoUpdateCfg
Friendly Name	Configure Automatic Updates
Element Name	Scheduled install day.
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage end user experience
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate\AU
ADMX File Name	WindowsUpdate.admx

ScheduledInstallEveryWeek

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1709 [10.0.16299] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/ScheduledInstallEveryWeek
```

Enables the IT admin to schedule the update installation on the every week. Value type is integer.

Note

This policy will only take effect if Update/AllowAutoUpdate has been configured to option 3 or 4 for scheduled installation.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	1

Allowed values:

Expand table

Value	Description
0	No update in the schedule.
1 (Default)	Update is scheduled every week.

Group policy mapping:

Expand table

Name	Value
Name	AutoUpdateCfg
Friendly Name	Configure Automatic Updates
Element Name	Every week.
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage end user experience
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate\AU
ADMX File Name	WindowsUpdate.admx

ScheduledInstallFirstWeek

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1709 [10.0.16299] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/ScheduledInstallFirstWeek
```

Enables the IT admin to schedule the update installation on the first week of the month. Value type is integer.

The ScheduledInstall*week policies operate on numeric days.

- ScheduledInstallFirstWeek: First week of the month (Days 1-7).
- ScheduledInstallSecondWeek: Second week of the month (Days 8-14).
- ScheduledInstallThirdWeek: Third week of the month (Days 15-21).
- ScheduledInstallFourthWeek: Fourth week of the month (Days 22-31).

These policies aren't exclusive and can be used in any combination. Together with ScheduledInstallDay, it defines the ordinal number of a weekday in a month. E.g., ScheduledInstallSecondWeek + ScheduledInstallDay = 3 is second Tuesday of the month. If the device is unavailable at the scheduled time, it can postpone installation of updates until the next month.

> **Note**
>
> This policy will only take effect if Update/AllowAutoUpdate has been configured to option 3 or 4 for scheduled installation.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace

Property name	Property value
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	No update in the schedule.
1	Update is scheduled every first week of the month.

Group policy mapping:

Expand table

Name	Value
Name	AutoUpdateCfg
Friendly Name	Configure Automatic Updates
Element Name	First week of the month.
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage end user experience
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate\AU
ADMX File Name	WindowsUpdate.admx

ScheduledInstallFourthWeek

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1709 [10.0.16299] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/ScheduledInstallFourthWeek
```

Enables the IT admin to schedule the update installation on the fourth week of the month. Value type is integer.

The ScheduledInstall*week policies operate on numeric days.

- ScheduledInstallFirstWeek: First week of the month (Days 1-7).
- ScheduledInstallSecondWeek: Second week of the month (Days 8-14).
- ScheduledInstallThirdWeek: Third week of the month (Days 15-21).
- ScheduledInstallFourthWeek: Fourth week of the month (Days 22-31).

These policies aren't exclusive and can be used in any combination. Together with ScheduledInstallDay, it defines the ordinal number of a weekday in a month. E.g., ScheduledInstallSecondWeek + ScheduledInstallDay = 3 is second Tuesday of the month. If the device is unavailable at the scheduled time, it can postpone installation of updates until the next month.

> **Note**
>
> This policy takes effect only when Update/AllowAutoUpdate is configured with option 3 or 4 for scheduled installation.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	No update in the schedule.
1	Update is scheduled every fourth week of the month.

Group policy mapping:

Expand table

Name	Value
Name	AutoUpdateCfg
Friendly Name	Configure Automatic Updates
Element Name	Fourth week of the month.
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage end user experience
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate\AU
ADMX File Name	WindowsUpdate.admx

ScheduledInstallSecondWeek

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1709 [10.0.16299] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/ScheduledInstallSecondWeek
```

Enables the IT admin to schedule the update installation on the second week of the month. Value type is integer.

The ScheduledInstall*week policies operate on numeric days.

- ScheduledInstallFirstWeek: First week of the month (Days 1-7).
- ScheduledInstallSecondWeek: Second week of the month (Days 8-14).
- ScheduledInstallThirdWeek: Third week of the month (Days 15-21).
- ScheduledInstallFourthWeek: Fourth week of the month (Days 22-31).

These policies aren't exclusive and can be used in any combination. Together with ScheduledInstallDay, it defines the ordinal number of a weekday in a month. E.g., ScheduledInstallSecondWeek + ScheduledInstallDay = 3 is second Tuesday of the month. If the device is unavailable at the scheduled time, it can postpone installation of updates until the next month.

> **Note**
>
> This policy takes effect only when Update/AllowAutoUpdate is configured with option 3 or 4 for scheduled installation.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	No update in the schedule.
1	Update is scheduled every second week of the month.

Group policy mapping:

Expand table

Name	Value
Name	AutoUpdateCfg
Friendly Name	Configure Automatic Updates
Element Name	Second week of the month.
Location	Computer Configuration

Name	Value
Path	Windows Components > Windows Update > Manage end user experience
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate\AU
ADMX File Name	WindowsUpdate.admx

ScheduledInstallThirdWeek

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1709 [10.0.16299] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/ScheduledInstallThirdWeek
```

Enables the IT admin to schedule the update installation on the third week of the month. Value type is integer.

The ScheduledInstall*week policies operate on numeric days.

- ScheduledInstallFirstWeek: First week of the month (Days 1-7).
- ScheduledInstallSecondWeek: Second week of the month (Days 8-14).
- ScheduledInstallThirdWeek: Third week of the month (Days 15-21).
- ScheduledInstallFourthWeek: Fourth week of the month (Days 22-31).

These policies aren't exclusive and can be used in any combination. Together with ScheduledInstallDay, it defines the ordinal number of a weekday in a month. E.g., ScheduledInstallSecondWeek + ScheduledInstallDay = 3 is second Tuesday of the month. If the device is unavailable at the scheduled time, it can postpone installation of updates until the next month.

> **Note**
>
> This policy takes effect only when Update/AllowAutoUpdate is configured with option 3 or 4 for scheduled installation.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	No update in the schedule.
1	Update is scheduled every third week of the month.

Group policy mapping:

Expand table

Name	Value
Name	AutoUpdateCfg
Friendly Name	Configure Automatic Updates
Element Name	Third week of the month.
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage end user experience
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate\AU
ADMX File Name	WindowsUpdate.admx

ScheduledInstallTime

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1507 [10.0.10240] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/ScheduledInstallTime
```

Enables the IT admin to schedule the time of the update installation. The data type is a integer. Supported values are 0-23, where 0 = 12 AM and 23 = 11 PM. The default value is 3.

> **Note**
>
> - This policy takes effect only when Update/AllowAutoUpdate is configured with option 3 or 4 for scheduled installation.
> - There's a window of approximately 30 minutes to allow for higher success rates of installation.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Allowed Values	Range: `[0-23]`
Default Value	3

Group policy mapping:

Expand table

Name	Value
Name	AutoUpdateCfg
Friendly Name	Configure Automatic Updates

Name	Value
Element Name	Scheduled install time.
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage end user experience
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate\AU
ADMX File Name	WindowsUpdate.admx

SetDisablePauseUXAccess

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1809 [10.0.17763] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/SetDisablePauseUXAccess
```

This setting allows to remove access to "Pause updates" feature.

Once enabled user access to pause updates is removed.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
1	Enable.
0 (Default)	Disable.

Group policy mapping:

Expand table

Name	Value
Name	DisablePauseUXAccess
Friendly Name	Remove access to "Pause updates" feature
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage end user experience
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
Registry Value Name	SetDisablePauseUXAccess
ADMX File Name	WindowsUpdate.admx

SetDisableUXWUAccess

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1809 [10.0.17763] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/SetDisableUXWUAccess
```

This setting allows you to remove access to scan Windows Update.

If you enable this setting user access to Windows Update scan, download and install is removed.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	Disabled.
1	Enabled.

Group policy mapping:

Expand table

Name	Value
Name	DisableUXWUAccess
Friendly Name	Remove access to use all Windows Update features
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage end user experience
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
Registry Value Name	SetDisableUXWUAccess
ADMX File Name	WindowsUpdate.admx

SetEDURestart

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1703 [10.0.15063] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/SetEDURestart
```

Enabling this policy for EDU devices that remain on Carts overnight will skip power checks to ensure update reboots will happen at the scheduled install time.

Enabling this policy will restrict updates to download and install outside of Active Hours. Updates are allowed to start even if there's a signed-in user or the device is on battery power, providing there's more than 70% battery capacity. Windows will schedule the device to wake from sleep 1 hour after the ActiveHoursEnd time with a 60-minute random delay. Devices will reboot immediately after the updates are installed. If there are still pending updates, the device continues to retry every hour for 4 hours.

The following rules are followed regarding battery power:

- Above 70% - allowed to start work;
- Above 40% - allowed to reboot;
- Above 20% - allowed to continue work.

This setting overrides the install deferral behavior of AllowAutoUpdate.

These settings are designed for education devices that remain in carts overnight that are left in sleep mode. It isn't designed for 1:1 devices.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	Not configured.
1	Configured.

Group policy mapping:

Expand table

Name	Value
Name	SetEDURestart
Friendly Name	Update Power Policy for Cart Restarts
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage end user experience
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
Registry Value Name	SetEDURestart
ADMX File Name	WindowsUpdate.admx

UpdateNotificationLevel

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1809 [10.0.17763] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/UpdateNotificationLevel
```

0 (default) - Use the default Windows Update notifications 1 - Turn off all notifications, excluding restart warnings 2 - Turn off all notifications, including restart warnings.

This policy allows you to define what Windows Update notifications users see. This policy doesn't control how and when updates are downloaded and installed.

Important if you choose not to get update notifications and also define other Group policy so that devices aren't automatically getting updates, neither you nor device users will be aware of critical security, quality, or feature updates, and your devices may be at risk.

If you select "Apply only during active hours" in conjunction with Option 1 or 2, then notifications will only be disabled during active hours. You can set active hours by setting "Turn off auto-restart for updates during active hours" or allow the device to set active hours based on user behavior. To ensure that the device stays secure, a notification will still be shown if this option is selected once "Specify deadlines for automatic updates and restarts" deadline has been reached if configured, regardless of active hours.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	Use the default Windows Update notifications.
1	Turn off all notifications, excluding restart warnings.
2	Turn off all notifications, including restart warnings.

Group policy mapping:

Expand table

Name	Value
Name	UpdateNotificationLevel
Friendly Name	Display options for update notifications
Location	Computer Configuration
Path	Windows Components > Windows Update > Manage end user experience
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
Registry Value Name	SetUpdateNotificationLevel
ADMX File Name	WindowsUpdate.admx

Legacy Policies

AlwaysAutoRebootAtScheduledTimeMinutes

> **Note**
>
> This is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 11, version 22H2 [10.0.22621] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/AlwaysAutoRebootAtScheduledTimeMinutes
```

- If you enable this policy, a restart timer will always begin immediately after Windows Update installs important updates, instead of first notifying users on the login screen for at least two days.

The restart timer can be configured to start with any value from 15 to 180 minutes. When the timer runs out, the restart will proceed even if the PC has signed-in users.

- If you disable or don't configure this policy, Windows Update won't alter its restart behavior.

If the "No auto-restart with logged-on users for scheduled automatic updates installations" policy is enabled, then this policy has no effect.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Allowed Values	Range: `[15-180]`
Default Value	15

Group policy mapping:

Expand table

Name	Value
Name	AlwaysAutoRebootAtScheduledTime
Friendly Name	Always automatically restart at the scheduled time
Element Name	work (minutes)
Location	Computer Configuration
Path	Windows Components > Windows Update > Legacy Policies
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate\AU
ADMX File Name	WindowsUpdate.admx

AutoRestartDeadlinePeriodInDays

Note

This is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1607 [10.0.14393] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/AutoRestartDeadlinePeriodInDays
```

Specify the deadline before the PC will automatically restart to apply updates. The deadline can be set 2 to 14 days past the default restart date.

The restart may happen inside active hours.

If you disable or don't configure this policy, the PC will restart according to the default schedule.

Enabling either of the following two policies will override the above policy:

1. No auto-restart with logged-on users for scheduled automatic updates installations.
2. Always automatically restart at scheduled time.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Allowed Values	Range: `[2-30]`
Default Value	7

Group policy mapping:

Expand table

Name	Value
Name	AutoRestartDeadline
Friendly Name	Specify deadline before auto-restart for update installation
Element Name	Quality Updates (days)
Location	Computer Configuration
Path	Windows Components > Windows Update > Legacy Policies
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

AutoRestartDeadlinePeriodInDaysForFeatureUpdates

> **Note**
>
> This is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1809 [10.0.17763] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/AutoRestartDeadlinePeriodInDaysForFeatureUpdates
```

Specify the deadline before the PC will automatically restart to apply updates. The deadline can be set 2 to 14 days past the default restart date.

The restart may happen inside active hours.

If you disable or don't configure this policy, the PC will restart according to the default schedule.

Enabling either of the following two policies will override the above policy:

1. No auto-restart with logged-on users for scheduled automatic updates installations.
2. Always automatically restart at scheduled time.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Allowed Values	Range: `[2-30]`
Default Value	7

Group policy mapping:

Expand table

Name	Value
Name	AutoRestartDeadline
Friendly Name	Specify deadline before auto-restart for update installation
Element Name	Feature Updates (days)
Location	Computer Configuration
Path	Windows Components > Windows Update > Legacy Policies
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

AutoRestartNotificationSchedule

Note

This is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1703 [10.0.15063] and later

Device

`./Device/Vendor/MSFT/Policy/Config/Update/AutoRestartNotificationSchedule`

Enable this policy to specify when auto-restart reminders are displayed.

You can specify the amount of time prior to a scheduled restart to notify the user.

If you disable or don't configure this policy, the default period will be used.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	15

Allowed values:

Expand table

Value	Description
15 (Default)	15 Minutes.
30	30 Minutes.
60	60 Minutes.

Value	Description
120	120 Minutes.
240	240 Minutes.

Group policy mapping:

Expand table

Name	Value
Name	AutoRestartNotificationConfig
Friendly Name	Configure auto-restart reminder notifications for updates
Element Name	Period (min)
Location	Computer Configuration
Path	Windows Components > Windows Update > Legacy Policies
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

AutoRestartRequiredNotificationDismissal

> **Note**
>
> This is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1703 [10.0.15063] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/AutoRestartRequiredNotificationDismissal
```

Enable this policy to specify the method by which the auto-restart required notification is dismissed. When a restart is required to install updates, the auto-restart required notification is displayed. By default, the notification is automatically dismissed after 25 seconds.

The method can be set to require user action to dismiss the notification.

If you disable or don't configure this policy, the default method will be used.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	1

Allowed values:

Expand table

Value	Description
1 (Default)	Auto Dismissal.
2	User Dismissal.

Group policy mapping:

Expand table

Name	Value
Name	AutoRestartRequiredNotificationDismissal
Friendly Name	Configure auto-restart required notification for updates
Element Name	Method.
Location	Computer Configuration
Path	Windows Components > Windows Update > Legacy Policies

Name	Value
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

DeferUpdatePeriod

> **Note**
>
> This is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1607 [10.0.14393] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/DeferUpdatePeriod
```

> **Note**
>
> Don't use this policy for Windows 10, version 1607 or later devices. You can continue to use DeferUpdatePeriod for Windows 10, version 1511 devices.

Allows IT Admins to specify update delays for up to four weeks. Supported values are 0-4, which refers to the number of weeks to defer updates.

- If the **Specify intranet Microsoft update service location** policy is enabled, then the "Defer upgrades by", "Defer updates by" and "Pause Updates and Upgrades" settings have no effect.
- If the **Allow Telemetry** policy is enabled and the Options value is set to 0, then the "Defer upgrades by", "Defer updates by" and "Pause Updates and Upgrades" settings have no effect.

OS upgrade:

- Maximum deferral: Eight months
- Deferral increment: One month
- Update type/notes:
  - Upgrade - 3689BDC8-B205-4AF4-8D4A-A63924C5E9D5

Update:

- Maximum deferral: One month
- Deferral increment: One week
- Update type/notes: If a machine has Microsoft Update enabled, any Microsoft Updates in these categories will also observe Defer / Pause logic:
  - Security Update - 0FA1201D-4330-4FA8-8AE9-B877473B6441
  - Critical Update - E6CF1350-C01B-414D-A61F-263D14D133B4
  - Update Rollup - 28BC880E-0592-4CBF-8F95-C79B17911D5F
  - Service Pack - 68C5B0A3-D1A6-4553-AE49-01D3A7827828
  - Tools - B4832BD8-E735-4761-8DAF-37F882276DAB
  - Feature Pack - B54E7D24-7ADD-428F-8B75-90A396FA584F
  - Update - CD5FFD1E-E932-4E3A-BF74-18BF0B1BBD83
  - Driver - EBFC1FC5-71A4-4F7B-9ACA-3B9A503104A0

Other/can't defer:

- Maximum deferral: No deferral
- Deferral increment: No deferral
- Update type/notes: Any update category not enumerated above falls into this category. - Definition Update - E0789628-CE08-4437-BE74-2495B842F43B

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Allowed Values	Range: `[0-4]`
Default Value	0

Group policy mapping:

Expand table

Name	Value
Name	DeferUpgrade
Path	WindowsUpdate > AT > WindowsComponents > WindowsUpdateCat
Element Name	DeferUpdatePeriodId

DeferUpgradePeriod

> **Note**
>
> This is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1607 [10.0.14393] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/DeferUpgradePeriod
```

Allows IT Admins to specify additional upgrade delays for up to 8 months. Supported values are 0-8, which refers to the number of months to defer upgrades.

- If the **Specify intranet Microsoft update service location** policy is enabled, then the Defer upgrades by, Defer updates by and Pause Updates and Upgrades settings have no effect.
- If the **Allow Telemetry** policy is enabled and the Options value is set to 0, then the Defer upgrades by, Defer updates by and Pause Updates and Upgrades settings have no effect.

> **Note**

Don't use this policy for Windows 10, version 1607 or later devices. You can continue to use DeferUpgradePeriod for Windows 10, version 1511 devices.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Allowed Values	Range: `[0-8]`
Default Value	0

Group policy mapping:

Expand table

Name	Value
Name	DeferUpgrade
Path	WindowsUpdate > AT > WindowsComponents > WindowsUpdateCat
Element Name	DeferUpgradePeriodId

DisableDualScan

> **Note**
>
> This is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education	✅ Windows 10, version 1607 [10.0.14393] and later

Scope	Editions	Applicable OS
	✅ IoT Enterprise / IoT Enterprise LTSC	

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/DisableDualScan
```

Enable this policy to not allow update deferral policies to cause scans against Windows Update.

- If this policy is disabled or not configured, then the Windows Update client may initiate automatic scans against Windows Update while update deferral policies are enabled.

> **Note**
>
> This policy applies only when the intranet Microsoft update service this computer is directed to is configured to support client-side targeting. If the "Specify intranet Microsoft update service location" policy is disabled or not configured, this policy has no effect.

> **Note**
>
> For more information about dual scan, see Demystifying "Dual Scan" and Improving Dual Scan on 1607.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	Allow scan against Windows Update.

Value	Description
1	Don't allow update deferral policies to cause scans against Windows Update.

Group policy mapping:

Expand table

Name	Value
Name	DisableDualScan
Friendly Name	Do not allow update deferral policies to cause scans against Windows Update
Location	Computer Configuration
Path	Windows Components > Windows Update > Legacy Policies
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
Registry Value Name	DisableDualScan
ADMX File Name	WindowsUpdate.admx

EngagedRestartDeadline

> **Note**
>
> This is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1703 [10.0.15063] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/EngagedRestartDeadline
```

Enable this policy to control the timing before transitioning from Auto restarts scheduled outside of active hours to Engaged restart, which requires the user to schedule. The period can be set between 0 and 30 days from the time the restart becomes pending.

You can specify the number of days a user can snooze Engaged restart reminder notifications. The snooze period can be set between 1 and 3 days.

You can specify the deadline in days before automatically scheduling and executing a pending restart regardless of active hours. The deadline can be set between 2 and 30 days from the time the restart becomes pending. If configured, the pending restart will transition from Auto-restart to Engaged restart (pending user schedule) to automatically executed, within the specified period.

If you don't specify a deadline or if the deadline is set to 0, the PC won't automatically restart and will require the person to schedule it prior to restart.

If you disable or don't configure this policy, the PC will restart following the default schedule.

Enabling any of the following policies will override the above policy:

1. No auto-restart with logged-on users for scheduled automatic updates installations
2. Always automatically restart at scheduled time.
3. Specify deadline before auto-restart for update installation.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Allowed Values	Range: `[2-30]`
Default Value	14

Group policy mapping:

Expand table

Name	Value
Name	EngagedRestartTransitionSchedule

Name	Value
Friendly Name	Specify Engaged restart transition and notification schedule for updates
Element Name	Deadline (days)
Location	Computer Configuration
Path	Windows Components > Windows Update > Legacy Policies
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

EngagedRestartDeadlineForFeatureUpdates

> **Note**
>
> This is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1809 [10.0.17763] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/EngagedRestartDeadlineForFeatureUpdates
```

Enable this policy to control the timing before transitioning from Auto restarts scheduled outside of active hours to Engaged restart, which requires the user to schedule. The period can be set between 0 and 30 days from the time the restart becomes pending.

You can specify the number of days a user can snooze Engaged restart reminder notifications. The snooze period can be set between 1 and 3 days.

You can specify the deadline in days before automatically scheduling and executing a pending restart regardless of active hours. The deadline can be set between 2 and 30 days from the time the restart becomes pending. If configured, the pending restart will transition from Auto-

restart to Engaged restart (pending user schedule) to automatically executed, within the specified period.

If you don't specify a deadline or if the deadline is set to 0, the PC won't automatically restart and will require the person to schedule it prior to restart.

If you disable or don't configure this policy, the PC will restart following the default schedule.

Enabling any of the following policies will override the above policy:

1. No auto-restart with logged-on users for scheduled automatic updates installations
2. Always automatically restart at scheduled time.
3. Specify deadline before auto-restart for update installation.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Allowed Values	Range: `[2-30]`
Default Value	14

Group policy mapping:

Expand table

Name	Value
Name	EngagedRestartTransitionSchedule
Friendly Name	Specify Engaged restart transition and notification schedule for updates
Element Name	Deadline (days)
Location	Computer Configuration
Path	Windows Components > Windows Update > Legacy Policies
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

EngagedRestartSnoozeSchedule

> **Note**
>
> This is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1703 [10.0.15063] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/EngagedRestartSnoozeSchedule
```

Enable this policy to control the timing before transitioning from Auto restarts scheduled outside of active hours to Engaged restart, which requires the user to schedule. The period can be set between 0 and 30 days from the time the restart becomes pending.

You can specify the number of days a user can snooze Engaged restart reminder notifications. The snooze period can be set between 1 and 3 days.

You can specify the deadline in days before automatically scheduling and executing a pending restart regardless of active hours. The deadline can be set between 2 and 30 days from the time the restart becomes pending. If configured, the pending restart will transition from Auto-restart to Engaged restart (pending user schedule) to automatically executed, within the specified period.

If you don't specify a deadline or if the deadline is set to 0, the PC won't automatically restart and will require the person to schedule it prior to restart.

If you disable or don't configure this policy, the PC will restart following the default schedule.

Enabling any of the following policies will override the above policy:

1. No auto-restart with logged-on users for scheduled automatic updates installations
2. Always automatically restart at scheduled time.

3. Specify deadline before auto-restart for update installation.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Allowed Values	Range: `[1-3]`
Default Value	3

Group policy mapping:

Expand table

Name	Value
Name	EngagedRestartTransitionSchedule
Friendly Name	Specify Engaged restart transition and notification schedule for updates
Element Name	Snooze (days)
Location	Computer Configuration
Path	Windows Components > Windows Update > Legacy Policies
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

EngagedRestartSnoozeScheduleForFeatureUpdates

> **Note**
>
> This is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1809 [10.0.17763] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/EngagedRestartSnoozeScheduleForFeatureUpdates
```

Enable this policy to control the timing before transitioning from Auto restarts scheduled outside of active hours to Engaged restart, which requires the user to schedule. The period can be set between 0 and 30 days from the time the restart becomes pending.

You can specify the number of days a user can snooze Engaged restart reminder notifications. The snooze period can be set between 1 and 3 days.

You can specify the deadline in days before automatically scheduling and executing a pending restart regardless of active hours. The deadline can be set between 2 and 30 days from the time the restart becomes pending. If configured, the pending restart will transition from Auto-restart to Engaged restart (pending user schedule) to automatically executed, within the specified period.

If you don't specify a deadline or if the deadline is set to 0, the PC won't automatically restart and will require the person to schedule it prior to restart.

If you disable or don't configure this policy, the PC will restart following the default schedule.

Enabling any of the following policies will override the above policy:

1. No auto-restart with logged-on users for scheduled automatic updates installations
2. Always automatically restart at scheduled time.
3. Specify deadline before auto-restart for update installation.

Description framework properties:

Expand table

Property name	Property value
Format	`int`

Property name	Property value
Access Type	Add, Delete, Get, Replace
Allowed Values	Range: `[1-3]`
Default Value	3

Group policy mapping:

Expand table

Name	Value
Name	EngagedRestartTransitionSchedule
Friendly Name	Specify Engaged restart transition and notification schedule for updates
Element Name	Snooze (days)
Location	Computer Configuration
Path	Windows Components > Windows Update > Legacy Policies
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

EngagedRestartTransitionSchedule

> **Note**
>
> This is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1703 [10.0.15063] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/EngagedRestartTransitionSchedule
```

Enable this policy to control the timing before transitioning from Auto restarts scheduled outside of active hours to Engaged restart, which requires the user to schedule. The period can be set between 0 and 30 days from the time the restart becomes pending.

You can specify the number of days a user can snooze Engaged restart reminder notifications. The snooze period can be set between 1 and 3 days.

You can specify the deadline in days before automatically scheduling and executing a pending restart regardless of active hours. The deadline can be set between 2 and 30 days from the time the restart becomes pending. If configured, the pending restart will transition from Auto-restart to Engaged restart (pending user schedule) to automatically executed, within the specified period.

If you don't specify a deadline or if the deadline is set to 0, the PC won't automatically restart and will require the person to schedule it prior to restart.

If you disable or don't configure this policy, the PC will restart following the default schedule.

Enabling any of the following policies will override the above policy:

1. No auto-restart with logged-on users for scheduled automatic updates installations
2. Always automatically restart at scheduled time.
3. Specify deadline before auto-restart for update installation.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Allowed Values	Range: `[0-30]`
Default Value	7

Group policy mapping:

Expand table

Name	Value
Name	EngagedRestartTransitionSchedule
Friendly Name	Specify Engaged restart transition and notification schedule for updates
Element Name	Transition (days)
Location	Computer Configuration
Path	Windows Components > Windows Update > Legacy Policies
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

EngagedRestartTransitionScheduleForFeatureUpdates

> **Note**
>
> This is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1809 [10.0.17763] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/EngagedRestartTransitionScheduleForFeatureUpdates
```

Enable this policy to control the timing before transitioning from Auto restarts scheduled outside of active hours to Engaged restart, which requires the user to schedule. The period can be set between 0 and 30 days from the time the restart becomes pending.

You can specify the number of days a user can snooze Engaged restart reminder notifications. The snooze period can be set between 1 and 3 days.

You can specify the deadline in days before automatically scheduling and executing a pending restart regardless of active hours. The deadline can be set between 2 and 30 days from the time the restart becomes pending. If configured, the pending restart will transition from Auto-restart to Engaged restart (pending user schedule) to automatically executed, within the specified period.

If you don't specify a deadline or if the deadline is set to 0, the PC won't automatically restart and will require the person to schedule it prior to restart.

If you disable or don't configure this policy, the PC will restart following the default schedule.

Enabling any of the following policies will override the above policy:

1. No auto-restart with logged-on users for scheduled automatic updates installations
2. Always automatically restart at scheduled time.
3. Specify deadline before auto-restart for update installation.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Allowed Values	Range: `[0-30]`
Default Value	7

Group policy mapping:

Expand table

Name	Value
Name	EngagedRestartTransitionSchedule
Friendly Name	Specify Engaged restart transition and notification schedule for updates
Element Name	Transition (days)
Location	Computer Configuration
Path	Windows Components > Windows Update > Legacy Policies

Name	Value
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

IgnoreMOAppDownloadLimit

> **Note**
>
> This is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1703 [10.0.15063] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/IgnoreMOAppDownloadLimit
```

Specifies whether to ignore the MO download limit (allow unlimited downloading) over a cellular network for apps and their updates. If lower-level limits (for example, mobile caps) are required, those limits are controlled by external policies.

> **Warning**
>
> Setting this policy might cause devices to incur costs from MO operators.

To validate this policy:

1. Enable the policy and ensure the device is on a cellular network.
2. Run the scheduled task on your device to check for app updates in the background. For example, on a device, run the following commands in TShell:

TShell

```
exec-device schtasks.exe -arguments '/run /tn
"\Microsoft\Windows\WindowsUpdate\Automatic App Update" /I'
```

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	Don't ignore MO download limit for apps and their updates.
1	Ignore MO download limit (allow unlimited downloading) for apps and their updates.

IgnoreMOUpdateDownloadLimit

> **Note**
>
> This is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1703 [10.0.15063] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/IgnoreMOUpdateDownloadLimit
```

Specifies whether to ignore the MO download limit (allow unlimited downloading) over a cellular network for OS updates. If lower-level limits (for example, mobile caps) are required, those limits are controlled by external policies.

> ⚠ **Warning**
>
> Setting this policy might cause devices to incur costs from MO operators.

To validate this policy:

1. Enable the policy and ensure the device is on a cellular network.
2. Run the scheduled task on your device to check for app updates in the background. For example, on a device, run the following commands in TShell:

   TShell

   ```
   exec-device schtasks.exe -arguments '/run /tn
   "\Microsoft\Windows\WindowsUpdate\Automatic App Update" /I'
   ```

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	Don't ignore MO download limit for OS updates.
1	Ignore MO download limit (allow unlimited downloading) for OS updates.

PauseDeferrals

> **Note**
>
> This is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1607 [10.0.14393] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/PauseDeferrals
```

> **Note**
>
> Don't use this policy for Windows 10, version 1607 or later devices. You can continue to use PauseDeferrals for Windows 10, version 1511 devices. Allows IT Admins to pause updates and upgrades for up to 5 weeks. Paused deferrals will be reset after 5 weeks. If the Specify intranet Microsoft update service location policy is enabled, then the Defer upgrades by, Defer updates by and Pause Updates and Upgrades settings have no effect. If the Allow Telemetry policy is enabled and the Options value is set to 0, then the Defer upgrades by, Defer updates by and Pause Updates and Upgrades settings have no effect.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	Deferrals aren't paused.
1	Deferrals are paused.

Group policy mapping:

Expand table

Name	Value
Name	DeferUpgrade
Path	WindowsUpdate > AT > WindowsComponents > WindowsUpdateCat
Element Name	PauseDeferralsId

PhoneUpdateRestrictions

> **Note**
>
> This is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1703 [10.0.15063] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/PhoneUpdateRestrictions
```

This policy is deprecated. Use Update/RequireUpdateApproval instead.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Allowed Values	Range: `[0-4]`
Default Value	4

RequireDeferUpgrade

> **Note**
>
> This is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1607 [10.0.14393] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/RequireDeferUpgrade
```

> **Note**
>
> Don't use this policy for Windows 10, version 1607 or later devices. You can continue to use RequireDeferUpgrade for Windows 10, version 1511 devices. Allows the IT admin to set a device to Semi-Annual Enterprise Channel train.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	User gets upgrades from Semi-Annual Enterprise Channel (Preview).
1	User gets upgrades from Semi-Annual Enterprise Channel.

Group policy mapping:

Expand table

Name	Value
Name	DeferUpgrade
Path	WindowsUpdate > AT > WindowsComponents > WindowsUpdateCat
Element Name	DeferUpgradePeriodId

RequireUpdateApproval

> **Note**
>
> This is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Expand table

Scope	Editions	Applicable OS
✅ Device	✅ Pro ✅ Enterprise	✅ Windows 10, version 1507 [10.0.10240] and later

Scope	Editions	Applicable OS
❌ User	✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/RequireUpdateApproval
```

> **Note**
>
> If you previously used the Update/PhoneUpdateRestrictions policy in previous versions of Windows, it has been deprecated. Please use this policy instead. Allows the IT admin to restrict the updates that are installed on a device to only those on an update approval list. It enables IT to accept the End User License Agreement (EULA) associated with the approved update on behalf of the end-user. EULAs are approved once an update is approved.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	Not configured. The device installs all applicable updates.
1	The device only installs updates that are both applicable and on the Approved Updates list. Set this policy to 1 if IT wants to control the deployment of updates on devices, such as when testing is required prior to deployment.

ScheduleImminentRestartWarning

> **Note**
>
> This is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1703 [10.0.15063] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/ScheduleImminentRestartWarning
```

Enable this policy to control when notifications are displayed to warn users about a scheduled restart for the update installation deadline. Users aren't able to postpone the scheduled restart once the deadline has been reached and the restart is automatically executed.

Specifies the amount of time prior to a scheduled restart to display the warning reminder to the user.

You can specify the amount of time prior to a scheduled restart to notify the user that the auto restart is imminent to allow them time to save their work.

If you disable or don't configure this policy, the default notification behaviors will be used.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	15

Allowed values:

Expand table

Value	Description
15 (Default)	15 Minutes.
30	30 Minutes.
60	60 Minutes.

Group policy mapping:

Expand table

Name	Value
Name	RestartWarnRemind
Friendly Name	Configure auto-restart warning notifications schedule for updates
Element Name	Warning (mins)
Location	Computer Configuration
Path	Windows Components > Windows Update > Legacy Policies
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

ScheduleRestartWarning

> **Note**
>
> This is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education	✅ Windows 10, version 1703 [10.0.15063] and later

Scope	Editions	Applicable OS
	✅ IoT Enterprise / IoT Enterprise LTSC	

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/ScheduleRestartWarning
```

Enable this policy to control when notifications are displayed to warn users about a scheduled restart for the update installation deadline. Users aren't able to postpone the scheduled restart once the deadline has been reached and the restart is automatically executed.

Specifies the amount of time prior to a scheduled restart to display the warning reminder to the user.

You can specify the amount of time prior to a scheduled restart to notify the user that the auto restart is imminent to allow them time to save their work.

If you disable or don't configure this policy, the default notification behaviors will be used.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	4

Allowed values:

Expand table

Value	Description
2	2 Hours.
4 (Default)	4 Hours.
8	8 Hours.
12	12 Hours.
24	24 Hours.

Group policy mapping:

Expand table

Name	Value
Name	RestartWarnRemind
Friendly Name	Configure auto-restart warning notifications schedule for updates
Element Name	Reminder (hours)
Location	Computer Configuration
Path	Windows Components > Windows Update > Legacy Policies
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

SetAutoRestartNotificationDisable

> **Note**
>
> This is a legacy policy and isn't applicable for Windows 11. Legacy policies might be removed in a future release.

Expand table

Scope	Editions	Applicable OS
✅ Device ❌ User	✅ Pro ✅ Enterprise ✅ Education ✅ IoT Enterprise / IoT Enterprise LTSC	✅ Windows 10, version 1703 [10.0.15063] and later

Device

```
./Device/Vendor/MSFT/Policy/Config/Update/SetAutoRestartNotificationDisable
```

This policy setting allows you to control whether users receive notifications for auto restarts for update installations including reminder and warning notifications.

Enable this policy to turn off all auto restart notifications.

If you disable or don't configure this policy, the default notification behaviors will be unchanged.

Description framework properties:

Expand table

Property name	Property value
Format	`int`
Access Type	Add, Delete, Get, Replace
Default Value	0

Allowed values:

Expand table

Value	Description
0 (Default)	Enabled.
1	Disabled.

Group policy mapping:

Expand table

Name	Value
Name	AutoRestartNotificationDisable
Friendly Name	Turn off auto-restart notifications for update installations
Location	Computer Configuration
Path	Windows Components > Windows Update > Legacy Policies
Registry Key Name	Software\Policies\Microsoft\Windows\WindowsUpdate
ADMX File Name	WindowsUpdate.admx

Related articles

Policy configuration service provider

Last updated on 08/14/2025

Update other Microsoft products

Article • 06/26/2024 • Applies to: ✅ Windows 11, ✅ Windows 10

This article contains a list of other Microsoft products that might be updated when the following policy is used:

- **Group policy**: Computer Configuration\Administrative Templates\Windows Components\Windows Update\Manage end user experience\Configure Automatic Updates
  - `Install updates for other Microsoft products` element under Configure Automatic Updates
- **MDM**: ./Device/Vendor/MSFT/Policy/Config/Update/AllowMUUpdateService

> **Note**
>
> This policy includes drivers. If you need to exclude drivers, use ExcludeWUDriversInQualityUpdate.

List of other Microsoft products

The following is a list of other Microsoft products that might be updated:

- Active Directory Rights Management Service
- AppFabric
- Azure File Sync
- Bing Bar
- Bing IME
- BizTalk
- East Asia IME for Windows Content
- Exchange
- IME Dictionary Update
- Live Search
- MED-V
- Microsoft Advanced Threat Analytics
- Microsoft Application Virtualization
- Microsoft Azure StorSimple
- Microsoft Configuration Manager
- Microsoft Dynamics CRM
- Microsoft Information Protection

- Microsoft Lync Server and Microsoft Lync
- Microsoft Monitoring Agent
- Microsoft SRS Device
- Microsoft StreamInsight
- Mobile and IoT
- MSRC
- .NET (also known as .NET Core)
- Office 2016 (MSI versions of Office)
- PlayReady
- Security intelligence updates for Microsoft Defender Antivirus and other Microsoft antimalware
- Silverlight
- Skype for Business
- SQL
- System Center Application Controller
- System Center Data Protection Manager
- System Center Operations Manager
- System Center Orchestrator
- System Center Virtual Machine Manager
- Visual Studio
- Windows Admin Center
- Windows Azure Hyper-V Recovery Manager
- Windows Azure Pack - Web Sites
- Windows Azure Pack
- Windows Azure Service Bus
- Windows Embedded Developer Update
- Windows Live Sign-in Assistant
- Windows Small Business Server
- Zune

Feedback

Was this page helpful? Yes No

Provide product feedback

Delivery Optimization reference

Applies to: ✅ Windows 11, ✅ Windows 10, ✅ Delivery Optimization

> **Looking for Group Policy objects?** See Delivery Optimization reference or the main spreadsheet available at the Download Center for Windows 11 or for Windows 10.

There are many configuration options you can set in Delivery Optimization to customize the content delivery experience specific to your environment needs. This article summarizes those configurations for your reference. If you just need an overview of Delivery Optimization, see What is Delivery Optimization. If you need information about setting up Delivery Optimization, including tips for the best settings in different scenarios, see Set up Delivery Optimization for Windows.

Delivery Optimization options

You can use Group Policy or an MDM solution like Intune to configure Delivery Optimization.

The Delivery Optimization settings in Group Policy are under **Configuration\Policies\Administrative Templates\Windows Components\Delivery Optimization**. In MDM, the same settings are under **.Vendor/MSFT/Policy/Config/DeliveryOptimization/**.

Summary of Delivery Optimization settings

Expand table

Group Policy setting	MDM setting	Supported from version	Notes
Download mode	DODownloadMode	1511	Default is configured to LAN(1). The Group Download mode (2) combined with Group ID, enables administrators to create custom device groups that share content between devices in the group.

Group Policy setting	MDM setting	Supported from version	Notes
Group ID	DOGroupID	1511	Used with Group Download mode. If not configured, check GroupIDSource. When GroupID or GroupIDSource policies aren't configured, the GroupID is defined as the AD Site (1), Authenticated domain SID (2) or Microsoft Entra tenant ID (5), in that order.
Select the source of Group IDs	DOGroupIDSource	1803	If not configured, check Group ID. When the GroupID or GroupIDSource policies aren't configured, the Group is defined as the AD Site (1), Authenticated domain SID (2) or Microsoft Entra tenant ID (5), in that order.
Select a method to restrict peer selection	DORestrictPeerSelectionBy	1803	Default isn't configured.
Minimum RAM (inclusive) allowed to use peer caching	DOMinRAMAllowedToPeer	1703	Default value is 4 GB.
Minimum disk size allowed to	DOMinDiskSizeAllowedToPeer	1703	Default value is 32 GB.

Group Policy setting	MDM setting	Supported from version	Notes
use peer caching			
Max cache age	DOMaxCacheAge	1511	Default value is 259,200 seconds (three days).
Max cache size	DOMaxCacheSize	1511	Default value is 20%.
Absolute max cache size (in GBs)	DOAbsoluteMaxCacheSize	1607	Default isn't configured.
Modify cache drive	DOModifyCacheDrive	1607	Default to the operating system drive through the %SYSTEMDRIVE% environment variable.
Minimum peer caching content file size	DOMinFileSizeToCache	1703	Default file size is 50 MB.
Monthly upload data cap	DOMonthlyUploadDataCap	1607	Default value is 20 GB.
Minimum background QoS	DOMinBackgroundQoS	1607	Default value is 20 MB/s.
Enable peer caching while the device connects via VPN	DOAllowVPNPeerCaching	1709	Default is to not allow peering while on VPN.
VPN keywords	DOVpnKeywords	Windows 11, version 22H2 with the September 2023 or later	Allows you to set one or more keywords used to recognize VPN connections.

Group Policy setting	MDM setting	Supported from version	Notes
		update installed	
Disallow cache server downloads from VPN	DODisallowCacheServerDownloadsOnVPN	Windows 11, version 22H2 with the September 2023 or later update installed	Disallow downloads from Microsoft Connected Cache servers when the device connects via VPN. By default, the device is allowed to download from Microsoft Connected Cache when connected via VPN.
Allow uploads while the device is on battery while under set battery level	DOMinBatteryPercentageAllowedToUpload	1709	Default is to not allow peering while on battery.
Maximum foreground download bandwidth (percentage)	DOPercentageMaxForegroundBandwidth	1803	Default is '0' which will dynamically adjust.
Maximum background download bandwidth (percentage)	DOPercentageMaxBackgroundBandwidth	1803	Default is '0' which will dynamically adjust.
Maximum foreground download bandwidth (in KB/s)	DOMaxForegroundDownloadBandwidth	2004	Default is '0' which will dynamically adjust.

Group Policy setting	MDM setting	Supported from version	Notes
Maximum background download bandwidth (in KB/s)	DOMaxBackgroundDownloadBandwidth	2004	Default is '0' which will dynamically adjust.
Set hours to limit foreground download bandwidth	DOSetHoursToLimitForegroundDownloadBandwidth	1803	Default isn't configured.
Set hours to limit background download bandwidth	DOSetHoursToLimitBackgroundDownloadBandwidth	1803	Default isn't configured.
Delay foreground download from HTTP (in secs)	DODelayForegroundDownloadFromHttp	1803	Default isn't configured. For peering, use this policy to delay the fallback to the HTTP source. Learn more about the different delay options.
Delay background download from HTTP (in secs)	DODelayBackgroundDownloadFromHttp	1803	Default isn't configured. For peering, use this policy to delay the fallback to the HTTP source. Learn more about the different delay options.
Delay foreground download cache server fallback (in secs)	DelayCacheServerFallbackForeground	1903	Default isn't configured. For Microsoft Connected Cache content, use this policy to delay the fallback to the HTTP source. Learn

Group Policy setting	MDM setting	Supported from version	Notes
			more about the different delay options.
Delay background download cache server fallback (in secs)	DelayCacheServerFallbackBackground	1903	Default isn't configured. For Microsoft Connected Cache content, use this policy to delay the fallback to the HTTP source. Learn more about the different delay options.
Cache server hostname	DOCacheHost	1809	No value is configured as default.
Cache server hostname source	DOCacheHostSource	2004	No value is configured as default.
Maximum download bandwidth	DOMaxDownloadBandwidth	1607 (deprecated in Windows 10, version 2004); use Maximum background download bandwidth (in KB/s) or Maximum foreground download bandwidth (in KB/s) instead)	Default is '0' which will dynamically adjust.
Percentage of maximum download bandwidth	DOPercentageMaxDownloadBandwidth	1607 (deprecated in Windows 10, version 2004); use Maximum	Default is '0' which will dynamically adjust.

Group Policy setting	MDM setting	Supported from version	Notes
		background download bandwidth (in KB/s) or Maximum foreground download bandwidth (in KB/s) instead)	
Maximum upload bandwidth	DOMaxUploadBandwidth	1607 (deprecated in Windows 10, version 2004)	Default is '0' (unlimited).

More detail on Delivery Optimization settings

Locally cached updates

Delivery Optimization uses locally cached updates to deliver contact via peers. The more content available in the cache, the more likely that peering can be used. In cases where devices have enough local storage and you'd like to cache more content. Likewise, if you have limited storage and would prefer to cache less, use the following settings to adjust the Delivery Optimization cache to suit your scenario:

- DOMaxCacheSize and DOAbsoluteMaxCacheSize control the amount of space the Delivery Optimization cache can use.
- DOMaxCacheAge controls the retention period for each update in the cache.
- The system drive is the default location for the Delivery Optimization cache. DOModifyCacheDrive allows administrators to change that location.

> **Note**
>
> It is possible to configure preferred cache devices. For more information, see DOGroupID.

All cached files have to be above a set minimum size. This size is automatically set by the Delivery Optimization cloud services, but when local storage is sufficient and the network isn't strained or congested, administrators might choose to change it to obtain increased

performance. You can set the minimum size of files to cache by adjusting DOMinFileSizeToCache.

Impact to network

More options available that control the impact Delivery Optimization has on your network include the following settings:

- Minimum Background QoS lets administrators guarantee a minimum download speed for Windows updates. This setting adjusts the amount of data downloaded directly from HTTP sources, rather than other peers in the network.
- Maximum Foreground Download Bandwidth specifies the maximum foreground download bandwidth that Delivery Optimization uses, across all concurrent download activities, as a percentage of available download bandwidth.
- Maximum Background Download Bandwidth specifies the maximum background download bandwidth that Delivery Optimization uses, across all concurrent download activities, as a percentage of available download bandwidth.
- Set Business Hours to Limit Background Download Bandwidth specifies the maximum background download bandwidth that Delivery Optimization uses during and outside business hours across all concurrent download activities as a percentage of available download bandwidth.
- Set Business Hours to Limit Foreground Download Bandwidth specifies the maximum foreground download bandwidth that Delivery Optimization uses during and outside business hours across all concurrent download activities as a percentage of available download bandwidth.
- Select a method to restrict Peer Selection restricts peer selection by the options you select.
- Select the source of Group IDs restricts peer selection to a specific source.

Policies to prioritize the use of peer-to-peer and cache server sources

When Delivery Optimization client is configured to use peers and Microsoft Connected Cache, to achieve the best possible content delivery experience, the client connects to both Connected Cache and peers in parallel. If the desired content can't be obtained from Connected Cache or peers, Delivery Optimization will automatically fall back to the HTTP source to get the requested content. There are four settings that allow you to prioritize peer-to-peer or Connected Cache sources by delaying the immediate fallback to HTTP source, which is the default behavior.

Peer-to-peer delay fallback settings

- Delay foreground download from HTTP (in secs) allows you to delay the use of an HTTP source in a foreground (interactive) download that is allowed to use P2P.
- Delay background download from HTTP (in secs) allows you to delay the use of an HTTP source in a background download that is allowed to use P2P.

Microsoft Connected Cache delay fallback settings

- Delay foreground download cache server fallback (in secs) allows you to delay the use of an HTTP source in a foreground (interactive) download that is allowed to use a cache server.
- Delay background download cache server fallback (in secs) allows you to delay the use of an HTTP source in a background download that is allowed to use a cache server.

If both peer-to-peer and Connected Cache are configured, the peer-to-peer delay settings will take precedence over the cache server delay settings. This setting allows Delivery Optimization to discover peers first then recognize the fallback setting for the Connected Cache cache server.

System resource usage

Administrators can further customize scenarios where Delivery Optimization is used with the following settings:

- Minimum RAM (inclusive) allowed to use Peer Caching configures the minimum RAM required for peer caching to be enabled.
- Minimum disk size allowed to use Peer Caching configures the minimum disk size required for peer caching to be enabled.
- Enable Peer Caching while the device connects via VPN allows clients connected through VPN to use peer caching.
- Allow uploads while the device is on battery while under set Battery level controls the minimum battery level required for uploads to occur. You must enable this policy to allow upload while on battery.

Download mode

MDM Setting: **DODownloadMode**

Download mode dictates which download sources clients are allowed to use when downloading Windows updates in addition to Windows Update servers. The following table shows the available download mode options and what they do. Other technical details for these policies are available in Policy CSP - Delivery Optimization.

Expand table

Download mode option	Functionality when configured
HTTP Only (0)	This setting disables peer-to-peer caching but still allows Delivery Optimization to download content over HTTP from the download's original source or a Microsoft Connected Cache server. This mode uses additional metadata provided by the Delivery Optimization cloud services for a peerless, reliable and efficient download experience.
LAN (**1 - Default**)	This default operating mode for Delivery Optimization enables peer sharing on the same network. The Delivery Optimization cloud service finds other clients that connect to the Internet using the same public IP as the target client. These clients then try to connect to other peers on the same network by using their private subnet IP.
Group (2)	When group mode is set, the group is automatically selected based on the device's Active Directory Domain Services (AD DS) site (Windows 10, version 1607) or the domain the device is authenticated to (Windows 10, version 1511). In group mode, peering occurs across internal subnets, between devices that belong to the same group, including devices in remote offices. You can use GroupID option to create your own custom group independently of domains and AD DS sites. Starting with Windows 10, version 1803, you can use the GroupIDSource parameter to take advantage of other method to create groups dynamically. Group download mode is the recommended option for most organizations looking to achieve the best bandwidth optimization with Delivery Optimization.
Internet (3)	Enable Internet peer sources for Delivery Optimization.
Simple (99)	Simple mode disables the use of Delivery Optimization cloud services completely (for offline environments). Delivery Optimization switches to this mode automatically when the Delivery Optimization cloud services are unavailable, unreachable, or when the content file size is less than 50 MB, as the default. In this mode, Delivery Optimization provides a reliable download experience over HTTP from the download's original source or a Microsoft Connected Cache server, with no peer-to-peer caching.
Bypass (100)	Starting in Windows 11, this option is deprecated. Don't configure **Download mode** to '100' (Bypass), which can cause some content to fail to download. If you want to disable peer-to-peer functionality, configure DownloadMode to (0). If your device doesn't have internet access, configure Download Mode to (99). When you configure Bypass (100), the download bypasses Delivery Optimization and uses BITS instead. You don't need to configure this option if you're using Configuration Manager.

Note

When you use Microsoft Entra tenant, AD Site, or AD Domain as the source of group IDs, the association of devices participating in the group should not be relied on for an authentication of identity of those devices.

Group ID

MDM Setting: **DOGroupID**

By default, peer sharing on clients using the Group download mode (option 2) is limited to the same domain in Windows 10, version 1511, and the same domain and Active Directory Domain Services site in Windows 10, version 1607. By using the Group ID setting, you can optionally create a custom group that contains devices that should participate in Delivery Optimization but don't fall within those domain or Active Directory Domain Services site boundaries, including devices in another domain. Using Group ID, you can further restrict the default group (for example, you could create a subgroup representing an office building), or extend the group beyond the domain, allowing devices in multiple domains in your organization to be peers. This setting requires the custom group to be specified as a GUID on each device that participates in the custom group.

> **Note**
>
> To generate a GUID using Powershell, use **[guid]::NewGuid()**
>
> This configuration is optional and not required for most implementations of Delivery Optimization.

Select the source of Group IDs

MDM Setting: **DOGroupIDSource**

Starting in Windows 10, version 1803, configure this policy to restrict peer selection to a specific source, when using a GroupID policy. The options are:

- 0 = Not set
- 1 = AD Site
- 2 = Authenticated domain SID
- 3 = DHCP Option ID (with this option, the client queries DHCP Option ID 234 and use the returned GUID value as the Group ID)
- 4 = DNS Suffix
- 5 = Starting with Windows 10, version 1903, you can use the Microsoft Entra tenant ID as a means to define groups. To do this configure the value for DOGroupIdSource to its new maximum value of 5.

When configured, the Group ID will be assigned automatically from the selected source. This policy is ignored if the GroupID policy is also configured. The default behavior, when the GroupID or GroupIDSource policies aren't configured, is to determine the Group ID using AD

Site (1), Authenticated domain SID (2) or Microsoft Entra tenant ID (5), in that order. If GroupIDSource is configured to either DHCP Option ID (3) or DNS Suffix (4) and those methods fail, the default behavior is used instead. The option configured in this policy only applies to Group (2) download mode. If Group (2) isn't configured as Download mode, this policy will be ignored. If you configure the value to anything other than 0-5, the policy is ignored.

Minimum RAM (inclusive) allowed to use peer caching

MDM Setting: **DOMinRAMAllowedToPeer**

This setting specifies the minimum RAM size in GB required to use Peer Caching. For example if the minimum set is 1 GB, then devices with 1 GB or higher available RAM will be allowed to use Peer caching. The recommended values are 1 to 4, and **the default value is 4 GB**.

Minimum disk size allowed to use peer caching

MDM Setting: **DOMinDiskSizeAllowedToPeer**

This setting specifies the required minimum disk size (capacity in GB) for the device to use Peer Caching. The recommended values are 64 to 256, and **the default value is 32 GB**.

> ⓘ **Note**
>
> If the Modify Cache Drive policy is configured, the disk size check applies to the new working directory specified by this policy.

Max cache age

MDM Setting: **DOMaxCacheAge**

In environments configured for Delivery Optimization, you might want to configure an expiration on cached updates and Windows application installation files. If so, this setting defines the maximum number of seconds each file can be held in the Delivery Optimization cache on each Windows 10 client device. Alternatively, organizations might choose to configure this value to "0" which means "unlimited" to avoid peers redownloading content. When "Unlimited" value is configured, Delivery Optimization holds the files in the cache longer and cleans up the cache as needed (for example when the cache size exceeded the maximum space allowed). **The default value is 259,200 seconds (three days)**.

Max cache size

MDM Setting: **DOMaxCacheSize**

This setting limits the maximum amount of space the Delivery Optimization cache can use as a percentage of the available drive space, from 1 to 100. For example, if you configure this value to 10 on a Windows client device that has 100 GB of available drive space, then Delivery Optimization uses up to 10 GB of that space. Delivery Optimization constantly assesses the available drive space and automatically clear the cache to keep the maximum cache size under the configured percentage. **The default value is 20%**.

Absolute max cache size

MDM Setting: **DOAbsoluteMaxCacheSize**

This setting specifies the maximum number of gigabytes the Delivery Optimization cache can use. This is different from the Max Cache Size setting, which is a percentage of available disk space. Also, if you configure this policy, it overrides the Max Cache Size setting. **The default value is not configured**.

Minimum peer caching content file size

MDM Setting: **DOMinFileSizeToCache**

This setting specifies the minimum content file size in MB enabled to use Peer Caching. The recommended values are from 1 to 100000. **The default file size is 50 MB** to participate in peering.

Maximum foreground download bandwidth

MDM Setting: **DOPercentageMaxForegroundBandwidth**

Starting in Windows 10, version 1803, specifies the maximum foreground download bandwidth that Delivery Optimization uses across all concurrent download activities as a percentage of available download bandwidth. **The default value of "0"** means that Delivery Optimization dynamically adjusts to use the available bandwidth for foreground downloads. However, downloads from LAN peers aren't throttled even when this policy is configured.

Maximum background download bandwidth

MDM Setting: **DOPercentageMaxBackgroundBandwidth**

Starting in Windows 10, version 1803, specifies the maximum background download bandwidth that Delivery Optimization uses across all concurrent download activities as a percentage of available download bandwidth. **The default value of "0"** means that Delivery Optimization dynamically adjusts to use the available bandwidth for background downloads. However, downloads from LAN peers aren't throttled even when this policy is configured.

> ⓘ **Note**
>
> It is recommended to use the absolute value download options 'DOMaxBackgroundDownloadBandwidth' and 'DOMaxForegroundDownloadBandwidth', rather than percentage-based options, for low bandwidth environments.

Set business hours to limit foreground download bandwidth

MDM Setting: **DOSetHoursToLimitForegroundDownloadBandwidth**

Starting in Windows 10, version 1803, specifies the maximum foreground download bandwidth that Delivery Optimization uses during and outside business hours across all concurrent download activities as a percentage of available download bandwidth. When DOSetHoursToLimitForegroundDownloadBandwidth is set, it will take precedence over the DOPercentageMaxForegroundBandwidth percentage values if set. **By default, this policy isn't configured.**

Set business hours to limit background download bandwidth

MDM Setting: **DOSetHoursToLimitBackgroundDownloadBandwidth**

Starting in Windows 10, version 1803, specifies the maximum background download bandwidth that Delivery Optimization uses during and outside business hours across all concurrent download activities as a percentage of available download bandwidth. When DOSetHoursToLimitBackgroundDownloadBandwidth is set, it will take precedence over the DOPercentageMaxBackgroundBandwidth percentage values if set. **By default, this policy isn't configured.**

Select a method to restrict peer selection

MDM Setting: **DORestrictPeerSelectionBy**

Starting in Windows 10, version 1803, configure this policy to further restrict peer selection in Download Modes LAN (1) and Group (2). In Windows 11, the 'Local Peer Discovery' option was introduced to restrict peer discovery to the local network. Currently the available options

include: 0 = None, 1 = Subnet mask, and 2 = Local Peer Discovery (DNS-SD). Choosing either Subnet mask (1) or Local Peer Discovery (2) will avoid peering between subnets.

If Group mode is configured, Delivery Optimization connects to locally discovered peers that are also part of the same Group (have the same Group ID) and prevents devices that aren't using the same Group ID from participating.

In Windows 11, the Local Peer Discovery (DNS-SD) option can be configured via MDM or Group Policy. However, in Windows 10, this feature can be enabled by setting the `HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\DeliveryOptimization\DORestrictPeerSelectionBy` value to **2**.

The default behaviors differ between Windows 10 and Windows 11. In Windows 10, there is no restriction configured. In Windows 11, the default peer selection is restricted to the Subnet only in LAN Download mode (1)

Delay foreground download from HTTP (in secs)

MDM Setting: **DODelayForegroundDownloadFromHttp**

Starting in Windows 10, version 1803, allows you to delay the use of an HTTP source in a foreground (interactive) download that is allowed to use peer-to-peer. **By default, this policy isn't configured.**

Delay background download from HTTP (in secs)

MDM Setting: **DODelayBackgroundDownloadFromHttp**

Starting in Windows 10, version 1803, this allows you to delay the use of an HTTP source in a background download that is allowed to use peer-to-peer. **By default, this policy isn't configured.**

Delay foreground download cache server fallback (in secs)

MDM Setting: **DelayCacheServerFallbackForeground**

Starting in Windows 10, version 1903, allows you to delay the fallback from cache server to the HTTP source for foreground content download by X seconds. If the 'Delay foreground download from HTTP policy is configured, it will apply first (to allow download from peers). **By default, this policy isn't configured.**

Delay background download cache server fallback (in secs)

MDM Setting: **DelayCacheServerFallbackBackground**

Starting in Windows 10, version 1903, configure this policy to delay the fallback from cache server to the HTTP source for a background content download by X seconds. If the 'Delay background download from HTTP' policy is configured, it will apply first (to allow downloads from peers) and then this policy will be applied. **By default, this policy isn't configured.**

Minimum background QoS

MDM Setting: **DOMinBackgroundQoS**

This policy controls background downloads and measures bandwidth between peers. When the measured speed falls below the configured threshold, content is downloaded from both peers and the HTTP source. Once the threshold is reached, HTTP connections are closed, and downloads continue only from peers.

Setting a lower threshold makes it easier to meet the limit, which means more content will come from peers and HTTP connections will not be used. **The default value is 20 MB/s.**

Modify cache drive

MDM Setting: **DOModifyCacheDrive**

This setting allows for an alternate Delivery Optimization cache location on the clients. **By default, the cache is stored on the operating system drive through the %SYSTEMDRIVE% environment variable.** You can configure the value to an environment variable (for example, %SYSTEMDRIVE%), a drive letter (for example, D:), or a folder path (for example, D:\DOCache).

Monthly upload data cap

MDM Setting: **DOMonthlyUploadDataCap**

This setting specifies the total amount of data in gigabytes that a Delivery Optimization client can upload to Internet peers per month. A value of "0" means that an unlimited amount of data can be uploaded. **The default value for this setting is 20 GB.**

Enable peer caching while the device connects via VPN

MDM Setting: **DOAllowVPNPeerCaching**

This setting determines whether a device will be allowed to participate in Peer Caching while connected to VPN. **By default, if a VPN connection is detected, peering isn't allowed, except**

when the 'Local Discovery' (DNS-SD) option is chosen. Specify "true" to allow the device to participate in Peer Caching while connected via VPN to the domain network. The device can download from or upload to other domain network devices, either on VPN or on the corporate domain network.

VPN keywords

MDM Setting: **DOVpnKeywords**

This policy allows you to configure one or more comma-separated keywords used to recognize VPN connections. **By default, this policy is not configured so if a VPN is detected, the device will not use peering.** Delivery Optimization automatically detects a VPN connection by looking at the network adapter's 'Description' and 'FriendlyName' strings using the default keyword list including: "VPN", "Secure", and "Virtual Private Network" (ex: "MSFTVPN" matches the "VPN" keyword). As the number of VPNs grow it's difficult to support an ever-changing list of VPN names. To address this, we've introduced this new setting to add unique VPN names to meet the needs of individual environments.

Disallow cache server downloads on VPN

MDM Setting: **DODisallowCacheServerDownloadsOnVPN**

This policy disallows downloads from Connected Cache servers when the device connects via VPN. **By default, the device is allowed to download from Connected Cache when connected via VPN.** Configure this policy if you prefer devices to download directly from the Internet when connected remotely (via VPN) instead of pulling from a Microsoft Connected Cache server deployed on your corporate network.

Allow uploads while the device is on battery while under set battery level

MDM Setting: **DOMinBatteryPercentageAllowedToUpload**

This setting specifies battery levels at which a device will be allowed to upload data. Specify any value between 1 and 100 (in percentage) to allow the device to upload data to LAN and Group peers while on DC power (Battery). Uploads will automatically pause when the battery level drops below the configured minimum battery level. The recommended value to configure, if you allow uploads on battery, is 40 (for 40%). The device can download from peers while on battery regardless of this policy.

> ⓘ **Important**

By default, devices will not upload while on battery. To enable uploads while on battery, you need to enable this policy and configure the battery value under which uploads pause.

Cache server hostname

MDM Setting: **DOCacheHost**

Configure this policy to designate one or more Microsoft Connected Cache servers to be used by Delivery Optimization. You can set one or more FQDNs or IP Addresses that are comma-separated, for example: myhost.somerandomhost.com,myhost2.somerandomhost.com,10.10.1.7. **By default, this policy has no value.** Delivery Optimization client will connect to the listed Microsoft Connected Cache servers in the order as they are listed. When multiple FQDNs or IP Addresses are listed, fallback to the CDN occurs immediately after the first failure in downloading from a cache server, unless the DelayCacheServerFallbackBackground or DelayCacheServerFallbackForeground policies are configured. When these delay policies are configured, the fallback occurs only after the configured delay time and the client continues to attempt connecting to the cache servers in round robin order before the delay time expires.

> **Important**
>
> Any value will signify that the policy is configured. For example, an empty string ("") isn't considered empty.

Cache server hostname source

MDM Setting: **DOCacheHostSource**

This policy allows you to specify how your client(s) can discover Delivery Optimization in Network Cache servers dynamically. There are two options:

- 1 = DHCP Option 235.
- 2 = DHCP Option 235 Force.

With either option, the client queries DHCP Option ID 235 and uses the returned value as the Cache Server Hostname. If DOCacheHost policy is also configured, then DHCP Option 235 Force (2) is required to override it. **By default, this policy has no value.**

Configure this policy to designate Delivery Optimization in Network Cache servers through a custom DHCP Option. Specify the custom DHCP option on your DHCP server as *text* type. You

can add one or more values as either fully qualified domain names (FQDN) or IP addresses. To add multiple values, separate each FQDN or IP address with commas.

> ⓘ **Note**
>
> If you format the DHCP Option ID incorrectly, the client will fall back to the Cache Server Hostname policy value if that value has been configured.
>
> If the LocalPolicyMerge setting is configured, such as part of security baselines, it can impact DHCP client and prevent it from retrieving this DHCP option, especially in Autopilot scenarios.

Maximum foreground download bandwidth (in KB/s)

MDM Setting: **DOMaxForegroundDownloadBandwidth**

Specifies the maximum foreground download bandwidth in kilobytes/second that the device can use across all concurrent download activities using Delivery Optimization. When this setting is enabled, either of the foreground percentage-based bandwidth limits — DOPercentageMaxForegroundBandwidth or DOSetHoursToLimitForegroundDownloadBandwidth - will apply as a percentage of the value specified here.

The default value of "0" means that Delivery Optimization dynamically adjusts to use the available bandwidth for downloads.

Maximum background download bandwidth (in KB/s)

MDM Setting: **DOMaxBackgroundDownloadBandwidth**

Specifies the maximum background download bandwidth in kilobytes/second that the device can use across all concurrent download activities using Delivery Optimization. When this setting is enabled, either of the background percentage-based bandwidth limits — DOPercentageMaxBackgroundBandwidth or DOSetHoursToLimitBackgroundDownloadBandwidth — will apply as a percentage of the value specified here.

The default value "0" means that Delivery Optimization dynamically adjusts to use the available bandwidth for downloads.

Deprecated policies

Maximum download bandwidth

MDM Setting: **DOMaxDownloadBandwidth**

Deprecated in Windows 10, version 2004. This setting specifies the maximum download bandwidth that can be used across all concurrent Delivery Optimization downloads in kilobytes per second (KB/s). **A default value of "0"** means that Delivery Optimization dynamically adjusts and optimizes the maximum bandwidth used.

Percentage of maximum download bandwidth

MDM Setting: **DOPercentageMaxDownloadBandwidth**

Deprecated in Windows 10, version 2004. This setting specifies the maximum download bandwidth that Delivery Optimization can use across all concurrent download activities as a percentage of available download bandwidth. **The default value of "0"** means that Delivery Optimization dynamically adjusts to use the available bandwidth for downloads.

Max upload bandwidth

MDM Setting: **DOMaxUploadBandwidth**

Deprecated in Windows 10, version 2004. This setting allows you to limit the number of upload bandwidth individual clients can use for Delivery Optimization. Consider this setting when clients are providing content to requesting peers on the network. This option is configured in kilobytes per second (KB/s). **The default value is "0" or "unlimited"** which means Delivery Optimization dynamically optimizes for minimal usage of upload bandwidth; however it doesn't cap the upload bandwidth rate at a set rate.

Last updated on 09/30/2025

How to make Features on Demand and language packs available when you're using WSUS or Configuration Manager

Article • 10/01/2024 •

Applies to: ✅ Windows 11, ✅ Windows 10, ✅ Microsoft Configuration Manager, ✅ WSUS

This reference article describes how to make Features on Demand (FoDs) and language packs available when you're using Windows Server Update Services (WSUS) or Configuration Manager for specific versions of Windows.

High-level changes affecting Features on Demand and language pack content

The following changes for FoD and language pack content affected how client policy needs to be configured:

- Starting in Windows 10 version 1709, you can't use WSUS to host Features on Demand (FoDs) locally.
- Starting with Windows 10 version 1803, language packs can no longer be hosted on WSUS.

Due to these changes, the **Specify settings for optional component installation and component repair** (ADMX_Servicing) policy, located under `Computer Configuration\Administrative Templates\System` was used to specify alternate ways to acquire FoDs and language packs, along with content for corruption repair. This policy allows specifying one alternate location. It's important to note the policy behaves differently across OS versions. For more information, see the Version specific information for Features on Demand and language packs section.

The introduction of the **Specify source service for specific classes of Windows Updates** (SetPolicyDrivenUpdateSourceFor<UpdateClass>) policy in Windows 10, version 2004 further complicated configuring settings for FoD and language pack content.

Starting in Windows 11, version 22H2, on-premises Unified Update Platform (UUP) updates were introduced. FoDs and language packs are available from WSUS again. It's no longer necessary to use the **Specify settings for optional component installation and component repair** policy for FoD and language pack content. This policy was modified starting in Windows 11, version 24H2 and the following options were removed:

- Never attempt to download payload from Windows Update
- Download repair content and optional features directly from Windows Update instead of Windows Server Update Services (WSUS)

Version specific information for Features on Demand and language packs

Windows 11, version 22H2, and later clients use on-premises Unified Update Platform (UUP) updates with WSUS and Microsoft Configuration Manager. These clients don't need to use **Specify settings for optional component installation and component repair** for FoDs and language packs since the content is available in WSUS due to on-premises UUP. The policy was modified starting in Windows 11, version 24H2 to remove the unneeded options.

For Windows 10, version 2004 through Windows 11, version 21H2, clients can't download FoDs or language packs when **Specify settings for optional component installation and component repair** is set to Windows Update and **Specify source service for specific classes of Windows Updates** (SetPolicyDrivenUpdateSourceFor<FeatureUpdates/QualityUpdates>) for either feature or quality updates is set to WSUS. If you need this content, you can set **Specify settings for optional component installation and component repair** to Windows Update and then either:

- Change the source selection for feature and quality updates to Windows Update
- Allow all classes of updates to come from WSUS by not configuring any source selections

> **Note**
>
> In Windows 10 version 21H2 and later, non-Administrator user accounts can add both a display language and its corresponding language features.

In Windows 10 version 1809 and later, changing the **Specify settings for optional component installation and component repair** policy also influences how language packs are acquired, however language packs can only be acquired directly from Windows Update (until Windows 11 version 22H2). It's currently not possible to acquire them from a network share. Specifying a network location works for FoD packages or corruption repair, depending on the content at that location.

In Windows 10 versions 1709 and 1803, changing the **Specify settings for optional component installation and component repair** policy to download content from

Windows Update enables acquisition of FoD packages while also enabling corruption repair. Specifying a network location works for either, depending on the content is found at that location. Changing this policy on these OS versions doesn't influence how language packs are acquired.

For all OS versions, changing the **Specify settings for optional component installation and component repair** policy doesn't affect how OS updates are distributed. They continue to come from WSUS, Configuration Manager, or other sources as you have scheduled them, even while optional content is sourced from Windows Update or a network location.

Learn about other client management options, including using Group Policy and administrative templates, in Manage Windows clients.

More resources

- WSUS documentation
- Configuration Manager documentation

Feedback

Was this page helpful? Yes No

Provide product feedback

How to add fonts that are missing after upgrading Windows

Article • 02/27/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

When Windows is upgraded from older versions of Windows to currently supported versions of Windows, previously available fonts might no longer be available post-upgrade. Many of the fonts that were previously shipped with Windows were moved to the optional features of Windows to:

- Reduce the operating system footprint.
- Improve performance.
- Optimize disk space usage.

In the following instances:

- A fresh instance of Windows is installed
- Windows is upgraded from an older version of Windows

optional fonts aren't enabled by default. As a result, these fonts are missing from the system. If documents were created using the missing fonts, these documents might display differently in the updated version of Windows.

For example, if an English, French, German, or Spanish version of Windows is installed, fonts such as the following are missing:

- `Gautami`
- `Meiryo`
- `Narkism/Batang`
- `BatangChe`
- `Dotum`
- `DotumChe`
- `Gulim`
- `GulimChe`
- `Gungsuh`
- `GungsuhChe`

If these fonts are needed, you can add them back to your system via optional features. The removal of these fonts is a permanent change in behavior for Windows, and it will remain this way in future releases.

Adding language-associated features via language settings

If fonts associated with a language are needed across the system, then that language should be added to the user profile followed by switching to that language. For example, the fonts are needed to:

- View web pages.
- Edit documents.
- Use apps in that language.

Adding a language and then switching to it can be done via the **Settings** app.

To add the fonts associated with a language and then switch to that language, first open the **Language & region** pane in the **Settings** app by selecting the following link:

Language & region

or

1. Right-click on the **Start** menu and select **Run**.

2. In the **Run** window, next to **Open:**, enter:

   Console

   ```
   ms-settings:regionlanguage
   ```

   and then select **OK**.

or

1. Right-click on the **Start** menu and select **Settings**.

2. In the left hand pane of the Settings app, select **Time & language**.

3. In the right hand **Time & language** pane, select **Language & region**.

Once the **Time & language** > **Language & region** pane is open, add the fonts associated with a language and then switch to that language with the following steps:

1. Under the **Language** section:

   a. Drop down the menu next to **Windows display language** to show all of the languages currently added to the device.

b. If the desired language isn't listed, add the fonts for the desired language by selecting the **Add a language** button next to **Preferred languages**.

c. In the **Choose a language to install** window that opens:

   i. Find the desired language by scrolling through the list of languages. Alternatively, search for the desired language in the **Type a language name...** search box.

   ii. Once the desired language is found, select the language, and then select the **Next** button.

   iii. In the **Install language features** window, select the desired options for the language, and then select the **Install** button.

d. The desired language should now be available in the drop-down menu next to **Windows display language**. Drop down the **Windows display language** menu and select the desired language.

e. Below **Windows display language** the message **Windows needs to sign you out in order to apply your new display language.** is displayed. Select the **Sign out** button to finish applying the language including the additional language fonts.

> ⓘ **Important**
>
> Windows Update is used to add the language including the associated fonts. The device needs to be online so Windows Update can download the content that it needs to add.

Add optional fonts manually without changing language settings

If fonts associated with a language are needed but aren't needed across the entire system, then that language should be added to the user profile as a supplemental font. Adding a font as a supplemental font doesn't require the user switch to that language. Adding a font as a supplemental font can be done via the **Settings** app.

To add supplemental fonts associated with a language while staying with the current language:

Open the **Optional features** pane in the **Settings** app by selecting the following link:

Optional features

or

1. Right-click on the **Start** menu and select **Run**.
2. In the **Run** window, next to **Open:**, enter:

   Console

   ```
   ms-settings:optionalfeatures
   ```

   and then select **OK**.

or

1. Right-click on the **Start** menu and select **Settings**.
2. In the left hand pane of the Settings app, select **System**.
3. In the right hand **System** pane, select **Optional features**.

> **Note**
>
> The navigation steps, UI elements, and UI text in this section are based on the latest version of Windows 11 with the latest cumulative update installed. For other versions of Windows 11 that are currently supported or don't have the latest cumulative update, some of the navigation steps, UI elements, and UI text might be different. For example, the Optional features pane might be located under **Settings** > **Apps**.

Once the **System** > **Optional features** pane is open, add a supplemental font with the following steps:

1. Select the **View features** button next to **Add an optional feature**.
2. In the **Add an optional feature** window that opens:
   a. Find the desired supplemental font to add and then select the box next to the supplemental font to add it. Multiple supplemental fonts can be selected.
   b. Once all of the desired supplemental fonts are selected, select the **Next** button.
   c. Review the selected list of features and then select the **Add** button to add the selected features.

> **ⓘ Important**
>
> Windows Update is used to add the optional features including the supplemental fonts. The device needs to be online so Windows Update can download the content that it needs to add.

Fonts included in optional font features

Here's a comprehensive list of the font families in each of the optional features. Some font families might include multiple fonts for different weights and styles.

- Arabic Script Supplemental Fonts: `Aldhabi, Andalus, Arabic Typesetting, Microsoft Uighur, Sakkal Majalla, Simplified Arabic, Traditional Arabic, Urdu Typesetting`
- Bangla Script Supplemental Fonts: `Shonar Bangla, Vrinda`
- Canadian Aboriginal Syllabics Supplemental Fonts: `Euphemia`
- Cherokee Supplemental Fonts: `Plantagenet Cherokee`
- Chinese (Simplified) Supplemental Fonts: `DengXian, FangSong, KaiTi, SimHei`
- Chinese (Traditional) Supplemental Fonts: `DFKai-SB, MingLiU, MingLiU_HKSCS, PMingLiU`
- Devanagari Supplemental Fonts: `Aparajita, Kokila, Mangal, Sanskrit Text, Utsaah`
- Ethiopic Supplemental Fonts: `Nyala`
- Gujarati Supplemental Fonts: `Shruti`
- Gurmukhi Supplemental Fonts: `Raavi`
- Hebrew Supplemental Fonts: `Aharoni Bold, David, FrankRuehl, Gisha, Levanim MT, Miriam, Miriam Fixed, Narkism, Rod`
- Japanese Supplemental Fonts: `Meiryo, Meiryo UI, MS Gothic, MS PGothic, MS UI Gothic, MS Mincho, MS PMincho, Yu Mincho`
- Kannada Supplemental Fonts: `Tunga`
- Khmer Supplemental Fonts: `DaunPenh, Khmer UI, MoolBoran`
- Korean Supplemental Fonts: `Batang, BatangChe, Dotum, DotumChe, Gulim, GulimChe, Gungsuh, GungsuhChe`
- Lao Supplemental Fonts: `DokChampa, Lao UI`
- Malayalam Supplemental Fonts: `Karthika`
- Odia Supplemental Fonts: `Kalinga`

- Pan-European Supplemental Fonts: `Arial Nova, Georgia Pro, Gill Sans Nova, Neue Haas Grotesk, Rockwell Nova, Verdana Pro`
- Sinhala Supplemental Fonts: `Iskoola Pota`
- Syriac Supplemental Fonts: `Estrangelo Edessa`
- Tamil Supplemental Fonts: `Latha, Vijaya`
- Telugu Supplemental Fonts: `Gautami, Vani`
- Thai Supplemental Fonts: `Angsana New, AngsanaUPC, Browallia New, BrowalliaUPC, Cordia New, CordiaUPC, DilleniaUPC, EucrosiaUPC, FreesiaUPC, IrisUPC, JasmineUPC, KodchiangUPC, Leelawadee, LilyUPC`

Related content

- Download the list of all available language FODs.
- Features On Demand V2 (Capabilities).
- Add Language Packs to Windows.

Feedback

Was this page helpful? Yes No

Provide product feedback

Windows Updates using forward and reverse differentials

Article • 08/28/2024 • Applies to: ✅ Windows 11, ✅ Windows 10

Windows monthly quality updates are cumulative, containing all previously released fixes to ensure consistency and simplicity. For an operating system platform like Windows, which stays in support for multiple years, the size of monthly quality updates can quickly grow large, thus directly impacting network bandwidth consumption.

Today, this problem is addressed by using express downloads, where differential downloads for every changed file in the update are generated based on selected historical revisions plus the base version. In this paper, we introduce a new technique to build compact software update packages that are applicable to any revision of the base version, and then describe how Windows quality updates use this technique.

General terms

The following general terms apply throughout this document:

- *Base version*: A major software release with significant changes, such as Windows 10, version 1809 (Windows 10 Build 17763.1)
- *Revision*: Minor releases in between the major version releases, such as KB4464330 (Windows 10 Build 17763.55)
- *Baseless Patch Storage Files (Baseless PSF)*: Patch storage files that contain full binaries or files

In this paper, we introduce a new technique that can produce compact software updates optimized for any origin/destination revision pair. It does this by calculating forward the differential of a changed file from the base version and its reverse differential back to the base version. Both forward and reverse differentials are then packaged as an update and distributed to the endpoints running the software to be updated. The update package contents can be symbolized as follows:

$$\Delta_{0 \rightarrow N}$$

$$\Delta_{N \rightarrow 0}$$

The endpoints that have the base version of the file (V_0) hydrate the target revision (V_N) by applying a simple transformation:

$$V_0 + \Delta_{0 \rightarrow N} = V_N$$

The endpoints that have revision N of the file (V_N), hydrate the target revision (V_R) by applying the following set of transformations:

$$V_N + \Delta_{N \rightarrow 0} = V_0$$

$$V_0 + \Delta_{0 \rightarrow R} = V_R$$

The endpoints retain the reverse differentials for the software revision they are on, so that it can be used for hydrating and applying next revision update.

By using a common baseline, this technique produces a single update package with numerous advantages:

- Compact in size
- Applicable to all baselines
- Simple to build
- Efficient to install
- Redistributable

Historically, download sizes of Windows quality updates (Windows 10, version 1803 and older supported versions of Windows 10) were optimized by using express download. Express download is optimized such that updating Windows systems download the minimum number of bytes. This is achieved by generating differentials for every

updated file based on selected historical base revisions of the same file + its base or RTM version.

For example, if the October monthly quality update has updated Notepad.exe, differentials for Notepad.exe file changes from September to October, August to October, July to October, June to October, and from the original feature release to October are generated. All these differentials are stored in a Patch Storage File (PSF, also referred to as express download files) and hosted or cached on Windows Update or other update management or distribution servers (for example, Windows Server Update Services (WSUS), Microsoft Configuration Manager, or a non-Microsoft update management or distribution server that supports express updates). A device applying express updates uses network protocol to determine optimal differentials, then downloads only what is needed from the update distribution endpoints.

The flip side of express download is that the size of PSF files can be large depending on the number of historical baselines against which differentials were calculated. Downloading and caching large PSF files to on-premises or remote update distribution servers is problematic for most organizations, hence they're unable to use express updates to keep their fleet of devices running Windows up to date. Secondly, due to the complexity of generating differentials and size of the express files that need to be cached on update distribution servers, it's only feasible to generate express download files for the most common baselines, thus express updates are only applicable to selected baselines. Finally, calculation of optimal differentials is expensive in terms of system memory utilization, especially for low-cost systems, impacting their ability to download and apply an update seamlessly.

In the following sections, we describe how quality updates use this technique based on forward and reverse differentials for newer releases of Windows and Windows Server to overcome the challenges with express downloads.

High-level design

Update packaging

Windows quality update packages contain forward differentials from quality update RTM baselines (ΔRTM→N) and reverse differentials back to RTM (ΔN→RTM) for each file that has changed since RTM. By using the RTM version as the baseline, we ensure that all devices have an identical payload. Update package metadata, content manifests, and forward and reverse differentials are packaged into a cabinet file (.cab). This .cab file, and the applicability logic, will also be wrapped in Microsoft Standalone Update (.msu) format.

There can be cases where new files are added to the system during servicing. These files won't have RTM baselines, thus forward and reverse differentials can't be used. In these scenarios, null differentials are used to handle servicing. Null differentials are the slightly compressed and optimized version of the full binaries. Update packages can have either forward or reverse differentials, or null differential of any given binary in them. The following image symbolizes the content of a Windows quality update installer:

Hydration and installation

Once the usual applicability checks are performed on the update package and are determined to be applicable, the Windows component servicing infrastructure hydrates the full files during preinstallation and then proceeds with the usual installation process.

Below is a high-level sequence of activities that the component servicing infrastructure runs in a transaction to complete installation of the update:

- Identify all files that are required to install the update.
- Hydrate each of necessary files using current version (V_N) of the file, reverse differential (V_N--->RTM) of the file back to quality update RTM/base version and forward differential (V_{RTM}--->R) from feature update RTM/base version to the target version. Also, use null differential hydration to hydrate null compressed files.
- Stage the hydrated files (full file), forward differentials (under `f` folder) and reverse differentials (under `r` folder) or null compressed files (under `n` folder) in the

component store (%windir%\WinSxS folder).
- Resolve any dependencies and install components.
- Clean up older state (V_{N-1}); the previous state V_N is retained for uninstallation and restoration or repair.

Resilient hydration

To ensure resiliency against component store corruption or missing files that could occur due to susceptibility of certain types of hardware to file system corruption, a corruption repair service has been traditionally used to recover the component store automatically (automatic corruption repair) or on demand (manual corruption repair) using an online or local repair source. This service will continue to offer the ability to repair and recover content for hydration and successfully install an update, if needed.

When corruption is detected during update operations, automatic corruption repair starts as usual and uses the Baseless Patch Storage File published to Windows Update for each update to fix corrupted manifests, binary differentials, or hydrated or full files. Baseless patch storage files contain reverse and forward differentials and full files for each updated component. Integrity of the repair files will be hash verified.

Corruption repair uses the component manifest to detect missing files and get hashes for corruption detection. During update installation, new registry flags for each differential staged on the machine are set. When automatic corruption repair runs, it scans hydrated files using the manifest and differential files using the flags. If the differential can't be found or verified, it's added to the list of corruptions to repair.

Lazy automatic corruption repair

"Lazy automatic corruption repair" runs during update operations to detect corrupted binaries and differentials. While applying an update, if hydration of any file fails, "lazy" automatic corruption repair automatically starts, identifies the corrupted binary or differential file, and then adds it to the corruption list. Later, the update operation continues as far as it can go, so that "lazy" automatic corruption repair can collect as many corrupted files to fix as possible. At the end of the hydration section, the update fails, and automatic corruption repair starts. Automatic corruption repair runs as usual and at the end of its operation, adds the corruption list generated by "lazy" automatic corruption repair on top of the new list to repair. Automatic corruption repair then repairs the files on the corruption list and installation of the update will succeed on the next attempt.

Feedback

Was this page helpful? Yes No

Provide product feedback

Customize Windows PE boot images

Applies to: ✅ Windows 11, ✅ Windows 10, ✅ Windows Server 2025, ✅ Windows Server 2022, ✅ Windows Server 2019, ✅ Windows Server 2016

The Windows PE (WinPE) boot images that are included with the Windows ADK have a minimal number of features and drivers. However the boot images can be customized by adding drivers, optional components, and applying the latest cumulative update.

Microsoft recommends updating Windows PE boot images with the latest cumulative update for maximum security and protection. The latest cumulative updates may also resolve known issues. For example, the Windows PE boot image can be updated with the latest cumulative update to address the BlackLotus UEFI bootkit vulnerability as documented in KB5025885: How to manage the Windows Boot Manager revocations for Secure Boot changes associated with CVE-2023-24932 and CVE-2023-24932.

> **Tip**
>
> The boot images from the ADK 10.1.26100.1 (May 2024) and later already contain the cumulative update to address the BlackLotus UEFI bootkit vulnerability.

This walkthrough describes how to customize a Windows PE boot image, including updating with the latest cumulative update, adding drivers, and adding optional components. Additionally this walkthrough goes over how customizations in boot images affect several different popular products that utilize boot images, such as Microsoft Configuration Manager, Microsoft Deployment Toolkit (MDT), and Windows Deployment Services (WDS).

Prerequisites

- Windows Assessment and Deployment Kit (Windows ADK) - It's recommended to use the latest version of the ADK.
- Windows PE add-on for the Windows ADK. Make sure the version of Windows PE matches the version of Windows ADK that is being used.
- Windows PE boot image.
- Latest cumulative update downloaded from the Microsoft Update Catalog site.

Steps

- Step 1: Download and install ADK
- Step 2: Download cumulative update (CU)

- Step 3: Backup existing boot image
- Step 4: Mount boot image to mount folder
- Step 5: Add drivers to boot image (optional)
- Step 6: Add optional components to boot image
- Step 7: Add cumulative update (CU) to boot image
- Step 8: Copy boot files from mounted boot image to ADK installation path
- Step 9: Perform component cleanup
- Step 10: Verify all desired packages have been added to boot image
- Step 11: Unmount boot image and save changes
- Step 12: Export boot image to reduce size
- Step 13: Update boot images in products that utilize the boot images (optional)

Step 1: Download and install ADK

1. Download and install the **Windows Assessment and Deployment Kit (Windows ADK)** from Download and install the Windows ADK.

   For this walk-through, when the Windows ADK is installed, it's only necessary to install the **Deployment Tools**. Other products, such as Microsoft Configuration Manager and Microsoft Deployment Toolkit (MDT), may require additional features installed, such as the **User State Migration Tool (USMT)**.

   One of the tools installed when installing the **Deployment Tools** feature is the **Deployment and Imaging Tools Environment** command prompt. When using the **Command Line** option to run the commands in this walk-through, make sure to run the commands from an elevated **Deployment and Imaging Tools Environment** command prompt. The **Deployment and Imaging Tools Environment** command prompt can be found in the Start Menu under **Windows Kits** > **Deployment and Imaging Tools Environment**.

   The paths in this article assume the Windows ADK was installed at the default location of `C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit`. If the Windows ADK was installed in a different location, then adjust the paths during the walk-through accordingly.

2. Download and install the **Windows PE add-on for the Windows ADK** from Download and install the Windows ADK. The **Windows PE add-on for the Windows ADK** is a separate download and install from the **Windows Assessment and Deployment Kit (Windows ADK)**. Make sure to individually download and install both.

> **Important**

It's strongly recommended to download and install the latest version of the Windows ADK and the Windows PE add-on for the Windows ADK.

In certain instances, older versions of the Windows ADK and Windows PE add-on may need to be used instead of the latest version. For example:

- Microsoft Deployment Toolkit (MDT) doesn't support versions of Windows or the Windows ADK beyond Windows 10. If using MDT, the recommendation is to instead use the ADK for Windows 10, version 2004. This version was the last version of the Windows ADK supported by MDT.
- The latest versions of the **Windows PE add-on for the Windows ADK** only includes a 64-bit boot image. If a 32-bit boot image is required, then the recommendation in this scenario is to also use the ADK for Windows 10, version 2004. This version of the Windows ADK was the last version to include both 32-bit and 64-bit boot images.

Step 2: Download cumulative update (CU)

1. Go to the Microsoft Update Catalog site and search for the latest cumulative update. The Windows version of the cumulative update should match the version of the Windows PE boot image that is being updated.
2. When searching the Microsoft Update Catalog site, use the search term `"<year>-<month> cumulative update for windows <x>"` where `year` is the four-digit current year, `<month>` is the two-digit current month, and `<x>` is the version of Windows that Windows PE is based on. Make sure to include the quotes (`"`). For example, to search for the latest cumulative update for Windows 11 in August 2023, use the search term `"2023-08 cumulative update for Windows 11"`, again making sure to include the quotes. If the cumulative update hasn't been released yet for the current month, then search for the previous month.

> **Tip**
>
> The boot images in the **ADK 10.1.25398.1 (September 2023)** are based off **Microsoft server operating system, version 22H2 for x64-based Systems**. Make sure to update the search term appropriately.

3. Once the cumulative update has been found, download the appropriate version for the version and architecture of Windows that matches the Windows PE boot image. For example, if the version of the Windows PE boot image is Windows 11 22H2 64-bit, then download the **Cumulative Update for Windows 11 Version 22H2 for x64-based Systems** version of the update.

4. Store the downloaded cumulative update in a known location for later use, for example `C:\Updates`.

> **Tip**
>
> It is recommended to use the full cumulative update when updating boot images with a cumulative update. However, instead of downloading the full cumulative update, the cumulative update for SafeOS can be downloaded and used instead. This will reduce the size of the final updated boot image. If any issues occur with a boot image updated with the SafeOS cumulative update, then use the full cumulative update instead.

> **Note**
>
> - When updating the boot image in the ADK 10.1.25398.1 (September 2023), download the **Cumulative Update for Microsoft server operating system version 23H2 for x64-based Systems**.
> - When updating the boot image in the ADK for Windows 10, version 2004, download the **Cumulative Update for Windows 10 Version 22H2**.

Step 3: Backup existing boot image

Before modifying the desired boot image, make a backup copy of the boot image that needs to be updated. For example:

- For the 64-bit boot image included with the **Windows PE add-on for the Windows ADK**, the boot image is located at `C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environment\amd64\en-us\winpe.wim`.

- For the default 64-bit boot image that is generated by **Microsoft Configuration Manager**, the boot image is located at `<ConfigMgr_Install_Directory>\OSD\boot\x64\boot.wim`. For other boot images in Configuration Manager, the path to the boot image is displayed in the **Image path:** field under the **Data Source** tab in the **Properties** of the boot image.

However, for **Microsoft Configuration Manager** it's recommended to instead modify the `winpe.wim` boot image included with the **Windows PE add-on for the Windows ADK**. For more information, see Microsoft Configuration Manager considerations.

- For the default 64-bit boot image that is generated by the **Microsoft Deployment Toolkit (MDT)**, the boot image is located at `<Deployment_Share>\Boot\LiteTouchPE_x64.wim`.

  However, for **Microsoft Deployment Toolkit (MDT)** it's recommended to instead modify the `winpe.wim` boot image included with the **Windows PE add-on for the Windows ADK**. For more information, see Microsoft Deployment Toolkit (MDT) considerations.

- For 64-bit boot images in **Windows Deployment Services (WDS)**, the boot images are located at `<RemoteInstall>\Boot\x64\Images`.

Adjust the above paths for 32-bit boot images (only available with Windows 10 ADKs).

The following command backs up the 64-bit boot image included with the **Windows PE add-on for the Windows ADK**:

PowerShell

From an elevated **PowerShell** command prompt, run the following command to create a backup copy of the 64-bit boot image included with the Windows ADK. If a backed-up boot image already exists, this command needs confirmation before it overwrites the existing backed-up boot image:

PowerShell

```
Copy-Item "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environment\amd64\en-us\winpe.wim" "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environment\amd64\en-us\winpe.bak.wim"
```

Adjust paths and file names accordingly to back up other boot images.

To automatically overwrite an existing backed-up boot image without confirmation, for example in a script, add the `-Force` parameter to the end of the command line.

For more information, see Copy-Item.

Important

When using the default `winpe.wim` boot image from the **Windows PE add-on for the Windows ADK**, it's recommended to always have a backed-up copy of the original

unmodified boot image. This allows reverting back to the pristine untouched original boot image in case any issues occur with any iteration of an updated boot image. Additionally, whenever a new cumulative update needs to be applied to a boot image, it's recommended to always start fresh and update from the original boot image with no updates instead of updating a previously updated boot image.

Step 4: Mount boot image to mount folder

1. Create a new empty folder to mount the boot image to. For example, `C:\Mount`. If using a previously created mount folder, ensure that it's empty and doesn't have any previously mounted images in it.

2. Mount the boot image to the mount folder using one of the following methods:

PowerShell

From an elevated **PowerShell** command prompt, run the following command to mount the boot image to the mount folder:

PowerShell

```
Mount-WindowsImage -Path "<Mount_folder_path>" -ImagePath "
<Boot_image_path>\<boot_image>.wim" -Index 1 -Verbose
```

Example:

PowerShell

```
Mount-WindowsImage -Path "C:\Mount" -ImagePath "C:\Program Files
(x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation
Environment\amd64\en-us\winpe.wim" -Index 1 -Verbose
```

For more information, see Mount-WindowsImage.

Step 5: Add drivers to boot image (optional)

If needed, add any drivers to the boot image:

PowerShell

From an elevated **PowerShell** command prompt, run one of the following commands to add drivers to the boot image:

PowerShell

```
Add-WindowsDriver -Path "<Mount_folder_path>" -Driver "<Driver_INF_source_path>\
<driver>.inf"
```

or

PowerShell

```
Add-WindowsDriver -Path "<Mount_folder_path>" -Driver "<Drivers_source_path>" -
Recurse
```

Examples:

PowerShell

```
Add-WindowsDriver -Path "C:\Mount" -Driver "C:\Drivers\driver.inf"
```

or

PowerShell

```
Add-WindowsDriver -Path "C:\Mount" -Driver "C:\Drivers" -Recurse
```

For more information, see Add-WindowsDriver.

The cumulative update installed later in this walkthrough doesn't affect drivers. Once a driver is added to a boot image, it doesn't need to be added again if a newer cumulative update is applied to the boot image.

Tip

A full set of drivers is not needed in Windows PE boot images. Only a small subset of drivers is needed that provides basic functionality while in WinPE. In most cases, no drivers need to be added to an out-of-box Windows ADK boot image since it already has many drivers built in. Don't add drivers to a boot image until it is verified that they are needed. When drivers do need to be added, generally only network (NIC) drivers are needed. Occasionally, mass storage (disk) may also be needed. Some Surface devices may also need keyboard and mouse drivers.

> **Important**
>
> For Microsoft Configuration Manager and Microsoft Deployment Toolkit (MDT) boot images, don't manually add drivers to the boot image using the above steps. Instead, add drivers to the boot images via Microsoft Configuration Manager or Microsoft Deployment Toolkit (MDT):
>
> - In Configuration Manager, via the **Drivers** tab in the **Properties** of the boot image.
> - In Microsoft Deployment Toolkit (MDT), via the **Drivers and Patches** tab under the **Windows PE** tab in the **Properties** of the deployment share.
>
> This will ensure that the drivers in the boot image can be properly managed through Configuration Manager or Microsoft Deployment Toolkit (MDT).

Step 6: Add optional components to boot image

1. Add any desired optional components to the boot image:

PowerShell

From an elevated **PowerShell** command prompt, run the following command to add optional components to the boot image:

PowerShell

```
Add-WindowsPackage -PackagePath "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environment\amd64\WinPE_OCs\<Component>.cab" -Path "<Mount_folder_path>" -Verbose
```

Example:

PowerShell

```
Add-WindowsPackage -PackagePath "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environment\amd64\WinPE_OCs\WinPE-Scripting.cab" -Path "C:\Mount" -Verbose
```

These examples assume a 64-bit boot image. If a different architecture is being used, then adjust the paths in the commands accordingly.

For more information, see Add-WindowsPackage.

2. After adding an optional component to the boot image, make sure to also add the language-specific component for that optional component.

   Not all optional components have the language-specific component. However, for optional components that do have a language-specific component, make sure that the language-specific component is installed.

   To check if an optional component has a language component, check the `C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environment\amd64\WinPE_OCs\<Language>\` directory to see if there's a matching language component for that optional component.

   For example, to install the English United States (en-us) language component for an optional component, use the following command line:

   PowerShell

   From an elevated **PowerShell** command prompt, run the following command to add the language components for the optional components to the boot image:

   PowerShell

   ```
   Add-WindowsPackage -PackagePath "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environment\amd64\WinPE_OCs\en-us\<Component>_en-us.cab" -Path "<Mount_folder_path>" -Verbose
   ```

   Example:

   PowerShell

   ```
   Add-WindowsPackage -PackagePath "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environment\amd64\WinPE_OCs\en-us\WinPE-Scripting_en-us.cab" -Path "C:\Mount" -Verbose
   ```

   These examples assume a 64-bit boot image. If a different architecture is being used, then adjust the paths accordingly.

   For more information, see Add-WindowsPackage.

ⓘ **Important**

When adding optional components, make sure to check if an optional component has a prerequisite for another optional component. When an optional component does have a prerequisite, make sure that the prerequisite component is installed first.

If an installation of an optional component that requires a prerequisite is attempted before installing the prerequisite, the following error message is displayed:

Windows Command Prompt

```
Error: 0x800f081e

The specified package is not applicable to this image.
```

For more information, see [WinPE Optional Components (OC) Reference: How to add Optional Components](#).

> **Important**
>
> Cumulative updates always need to be applied or reapplied after adding optional components to the boot image. If additional optional components are added to a boot image after a cumulative update has been applied, then the cumulative update needs to be reapplied.

> **Important**
>
> Both **Microsoft Configuration Manager** and **Microsoft Deployment Toolkit (MDT)** boot images require certain optional components to work properly. Make sure to add these required components when using either **Microsoft Configuration Manager** and **Microsoft Deployment Toolkit (MDT)**. For more information, see [Configuration Manager boot image required components](#) or [MDT boot image required components](#).
>
> Additionally, when adding any optional component for either **Microsoft Configuration Manager** or **Microsoft Deployment Toolkit (MDT)** boot images, make sure to manually add the optional components using this walkthrough instead of adding them through **Configuration Manager** or **MDT**. For more information and reasons why, see [Microsoft Configuration Manager considerations](#) or [Microsoft Deployment Toolkit (MDT) considerations](#).

Popular optional components

The following list contains the more popular optional components that are commonly added to boot images:

Expand table

Feature	File Name	Dependency	Purpose	Required by ConfigMgr	Required by MDT
Scripting/WinPE-Scripting	`WinPE-Scripting.cab`	NA	Supports running non-PowerShell scripts in WinPE	Yes	Yes
Network/WinPE-WDS-Tools	`WinPE-WDS-Tools.cab`	NA	Supports WDS in WinPE, including image capture and multicast	Yes	No
Scripting/WinPE-WMI	`WinPE-WMI.cab`	NA	Supports WMI and WMI scripting in WinPE	Yes	Yes
Startup/WinPE-SecureStartup	`WinPE-SecureStartup.cab`	Scripting/WinPE-WMI	Supports managing BitLocker and TPMs within WinPE	Yes	Yes
File management/WinPE-FMAPI	`WinPE-FMAPI.cab`	NA	Supports access to the Windows PE File Management API	No	Yes
Windows PowerShell/WinPE-PowerShell	`WinPE-PowerShell.cab`	Scripting/WinPE-Scripting Scripting/WinPE-WMI Microsoft .NET/WinPE-NetFx	Supports running PowerShell commands and scripts in WinPE	No	No
Microsoft .NET/WinPE-NetFx	`WinPE-NetFx.cab`	Scripting/WinPE-WMI	Supports .NET applications in	No	No

Feature	File Name	Dependency	Purpose	Required by ConfigMgr	Required by MDT
			WinPE		
Network/WinPE-Dot3Svc	`WinPE-Dot3Svc.cab`	NA	Supports the 802.1X network protocol in WinPE	No	No
HTML/WinPE-HTA	`WinPE-HTA.cab`	Scripting/WinPE-WMI	Supports running HTML applications in WinPE	No	No
Database/WinPE-MDAC	`WinPE-MDAC.cab`	NA	Supports connecting to databases in WinPE	No	No

For a full list of all available WinPE optional components including descriptions for each component, see WinPE Optional Components (OC) Reference: WinPE Optional Components.

Step 7: Add cumulative update (CU) to boot image

Apply the cumulative update (CU) downloaded during the Step 2: Download cumulative update (CU) step to the boot image:

PowerShell

From an elevated **PowerShell** command prompt, run the following command to add the cumulative update (CU) to the boot image:

PowerShell

```
Add-WindowsPackage -PackagePath "<Path_to_CU_MSU_update>\<CU>.msu" -Path "
<Mount_folder_path>" -Verbose
```

Example:

PowerShell

```
Add-WindowsPackage -PackagePath "C:\Updates\windows11.0-kb5029263-
x64_4f5fe19bbec786f5e445d3e71bcdf234fe2cbbec.msu" -Path "C:\Mount" -Verbose
```

For more information, see Add-WindowsPackage

> **Important**
>
> Make sure not to apply the cumulative update (CU) until all desired optional components have been installed via the Step 6: Add optional components to boot image step. Waiting to install the cumulative update (CU) until all optional components are installed makes sure that the optional components are also properly updated by the cumulative update. If in the future any additional optional components need to be added to the boot image, make sure to reapply the cumulative update.

Servicing stack update (SSU) and error 0x800f0823

Sometimes when applying a cumulative update (CU) to a boot image, you may receive error `0x800f0823`:

PowerShell

PowerShell

```
VERBOSE: Target Image Version <WinPE version>
WARNING: Failed to add package <Cumulative_Update_Path>\<Cumulative_Update>.msu
WARNING: Add-WindowsPackage failed. Error code = 0x800f0823
Add-WindowsPackage : An error occurred applying the Unattend.xml file from the
.msu package.
For more information, review the log file.
At line:1 char:1
+ Add-WindowsPackage -PackagePath "<Cumulative_Update_Path>\<Cumulative_Update>
...
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : NotSpecified: (:) [Add-WindowsPackage],
COMException
    + FullyQualifiedErrorId : Microsoft.Dism.Commands.AddWindowsPackageCommand
```

Inspecting the **DISM.log** reveals the following error:

PowerShell

PowerShell

```
Package "Package_for_RollupFix~<Cumulative_Update>" requires Servicing Stack
v<Required_Servicing_Stack_Version> but current Servicing Stack is
v<Current_Servicing_Stack_Version>. [HRESULT = 0x800f0823 -
CBS_E_NEW_SERVICING_STACK_REQUIRED]
Failed to initialize internal package [HRESULT = 0x800f0823 -
CBS_E_NEW_SERVICING_STACK_REQUIRED]
Failed to create internal package [HRESULT = 0x800f0823 -
CBS_E_NEW_SERVICING_STACK_REQUIRED]
Failed to create windows update package [HRESULT = 0x800f0823 -
CBS_E_NEW_SERVICING_STACK_REQUIRED]
DISM Package Manager: PID=<PID> TID=<TID> Failed opening package. -
CDISMPackageManager::Internal_CreatePackageByPath(hr:0x800f0823)
DISM Package Manager: PID=<PID> TID=<TID> Failed to get the underlying CBS
package. - CDISMPackageManager::OpenPackageByPath(hr:0x800f0823)
DISM Package Manager: PID=<PID> TID=<TID> The specified package cannot be added
to this Windows Image due to a version mismatch. - GetCbsErrorMsg
DISM Package Manager: PID=<PID> TID=<TID> Failed to open package at location
[<Temp_Path>\<Cumulative_Update>.cab]. -
CPackageManagerUnattendHandler::Internal_ProcessPackageFromSource(hr:0x800f0823)
DISM Package Manager: PID=<PID> TID=<TID> Failed to install package from source
[0] - trying next source location. hr = [0x800F0823] -
CPackageManagerUnattendHandler::Internal_UnattendInstallPackage
DISM Package Manager: PID=<PID> TID=<TID> Failed to Install the package
[Multiple_Packages~~~~0.0.0.0]. -
CPackageManagerUnattendHandler::Internal_UnattendInstallPackage(hr:0x800f0823)
DISM Package Manager: PID=<PID> TID=<TID> Package failed to install
[Multiple_Packages~~~~0.0.0.0]. -
CPackageManagerUnattendHandler::Internal_UnattendProcessPackage(hr:0x800f0823)
DISM Package Manager: PID=<PID> TID=<TID> Failed to process package at node
<package[1]>. - CPackageManagerUnattendHandler::Apply(hr:0x800f0823)
DISM Package Manager: PID=<PID> TID=<TID> Failed to Apply the unattend. -
CDISMPackageManager::Apply(hr:0x800f0823)
DISM Unattend Manager: PID=<PID> TID=<TID> "Error applying unattend for
provider: DISM Package Manager" - CUnattendManager::Apply(hr:0x800f0823)
DISM Package Manager: PID=<PID> TID=<TID> Failed applying the unattend file from
the MSU package. - CMsuPackage::ApplyMsuUnattend(hr:0x800f0823)
DISM Package Manager: PID=<PID> TID=<TID> Failed to apply the MSU unattend file
to the image. - CMsuPackage::Install(hr:0x800f0823)
API: PID=<PID> TID=<TID> Failed to install msu package <Path_to_CU_MSU_update>\
<MSU_Cumulative_Update>.msu -
CAddPackageCommandObject::InternalExecute(hr:0x800f0823)
API: PID=<PID> TID=<TID> InternalExecute failed -
CBaseCommandObject::Execute(hr:0x800f0823)
API: PID=<PID> TID=<TID> CAddPackageCommandObject internal execution failed -
DismAddPackageInternal(hr:0x800f0823)
```

The problem occurs when the WinPE boot image that is being serviced requires the installation of a servicing stack update (SSU) before installation of the cumulative update (CU) can occur. The problem usually occurs when using older Windows ADKs and older versions of Windows PE. The suggested fix is to upgrade to the latest version of the Windows ADK and Windows PE.

The latest versions of the Windows ADK and Windows PE most likely don't need a servicing stack update (SSU) installed before installing the cumulative update (CU).

For scenarios where older versions of the Windows ADK and Windows PE need to be used, for example when using Microsoft Deployment Toolkit (MDT), the servicing stack update needs to be installed before installing the cumulative update. The servicing stack update (SSU) is contained within the cumulative update (CU). To obtain the servicing stack update (SSU) so that it can be applied, it can be extracted from the cumulative update (CU).

The following steps outline how to extract and then install the servicing stack update (SSU) to the boot image. Once the servicing stack update (SSU) has been installed in the boot image, then the cumulative update (CU) should install to the boot image without error:

> **Important**
>
> These steps are only necessary if the error `0x800f0823` occurs when installing the cumulative update (CU) to the boot image. If error `0x800f0823` didn't occur when installing the cumulative update (CU) to the boot image, then skip to the next step [Step 8: Copy boot files from mounted boot image to ADK installation path]

1. Create a folder to extract the servicing stack update (SSU) into. For example, `C:\Updates\Extract`:

2. Extract the contents of the cumulative update (CU) to the folder created in the previous step using the following command:

   PowerShell

   PowerShell

   ```
   Start-Process "expand.exe" -ArgumentList " -f:* `"<Cumulative_Update_Path>\<Cumulative_Update>.msu`" `"<Extract_Folder_Path>`"" -Wait -LoadUserProfile
   ```

   Example:

   PowerShell

   ```
   Start-Process "expand.exe" -ArgumentList " -f:* `"C:\Updates\windows10.0-kb5028166-x64_fe3aa2fef685c0e76e1f5d34d529624294273f41.msu`" `"C:\Updates\Extract`"" -Wait -LoadUserProfile
   ```

   For more information, see Start-Process and expand.

3. Inspect the extracted files in the extract folder and identify the servicing stack update (SSU) CAB file. One of the files should be called `SSU-<Version>-<Arch>.cab`. For example, `SSU-19041.3205-x64.cab`. Make a note of the name of the servicing stack update (SSU) CAB file.

4. Using the name of the servicing stack update (SSU) CAB file obtained in the previous step, apply the servicing stack update (SSU) CAB file to the boot image using the following command:

   PowerShell

   From an elevated **PowerShell** command prompt, run the following command to add the cumulative update (CU) to the boot image:

   PowerShell

   ```
   Add-WindowsPackage -PackagePath "<Path_to_SSU_CAB_update>\<SSU>.cab" -Path "<Mount_folder_path>" -Verbose
   ```

   Example:

   PowerShell

   ```
   Add-WindowsPackage -PackagePath "C:\Updates\Extract\SSU-19041.3205-x64.cab" -Path "C:\Mount" -Verbose
   ```

   For more information, see Add-WindowsPackage.

5. Attempt to apply the cumulative update (CU) to the boot image again using the commands from Step 7: Add cumulative update (CU) to boot image.

Step 8: Copy boot files from mounted boot image to ADK installation path

Some cumulative updates contain updated bootmgr boot files that are added to the boot image. After these bootmgr boot files have been updated in the boot image, it's recommended to copy these updated bootmgr boot files from the boot image back to the Windows ADK. Copying these files ensures that the Windows ADK has the updated bootmgr boot files.

PowerShell

From an elevated **PowerShell** command prompt, run the following commands to copy the updated bootmgr boot files from the mounted boot image to the ADK installation path. These commands also back up any existing bootmgr boot files its finds. When applicable, the commands need confirmation to overwrite any existing files:

PowerShell

```
Copy-Item "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environment\amd64\Media\bootmgr.efi" "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environment\amd64\Media\bootmgr.bak.efi"

Copy-Item "<Mount_folder_path>\Windows\Boot\EFI\bootmgr.efi" "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environment\amd64\Media\bootmgr.efi"

Copy-Item "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environment\amd64\Media\EFI\Boot\bootx64.efi" "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environment\amd64\Media\EFI\Boot\bootx64.bak.efi"

Copy-Item "<Mount_folder_path>\Windows\Boot\EFI\bootmgfw.efi" "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environment\amd64\Media\EFI\Boot\bootx64.efi"
```

Example:

PowerShell

```
Copy-Item "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environment\amd64\Media\bootmgr.efi" "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environment\amd64\Media\bootmgr.bak.efi"

Copy-Item "C:\Mount\Windows\Boot\EFI\bootmgr.efi" "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environment\amd64\Media\bootmgr.efi"

Copy-Item "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environment\amd64\Media\EFI\Boot\bootx64.efi" "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environment\amd64\Media\EFI\Boot\bootx64.bak.efi"

Copy-Item "C:\Mount\Windows\Boot\EFI\bootmgfw.efi" "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environment\amd64\Media\EFI\Boot\bootx64.efi"
```

To overwrite the bootmgr boot files and any backed-up bootmgr boot file without confirmation, for example in a script, add the `-Force` parameter to the end of the

command lines.

For more information, see Copy-Item.

This step doesn't update or change the boot image. However, it makes sure that the latest bootmgr boot files are available to the Windows ADK when creating bootable media via the Windows ADK. When these files are updated in the Windows ADK, products that use the Windows ADK to create bootable media, such as **Microsoft Deployment Toolkit (MDT)**, also have access to the updated bootmgr boot files.

In particular, this step is needed when addressing the BlackLotus UEFI bootkit vulnerability as documented in KB5025885: How to manage the Windows Boot Manager revocations for Secure Boot changes associated with CVE-2023-24932 and CVE-2023-24932.

> **Tip**
>
> The boot images from the ADK 10.1.26100.1 (May 2024) and later already contain the cumulative update to address the BlackLotus UEFI bootkit vulnerability.

> **Note**
>
> **Microsoft Configuration Manager** and **Windows Deployment Services (WDS)** automatically extract the bootmgr boot files from the boot images when the boot images are updated in these products. They don't use the bootmgr boot files from the Windows ADK.

Step 9: Perform component cleanup

Run **DISM.exe** commands that clean up the mounted boot image and help reduce its size:

PowerShell

From an elevated **PowerShell** command prompt, run the following command to clean up the mounted boot image and help reduce its size:

PowerShell

```
Start-Process "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Deployment Tools\amd64\DISM\dism.exe" -ArgumentList " /Image:"<Mount_folder_path>" /Cleanup-image /StartComponentCleanup /Resetbase /Defer" -Wait -LoadUserProfile
```

```
Start-Process "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Deployment Tools\amd64\DISM\dism.exe" -ArgumentList " /Image:"<Mount_folder_path>" /Cleanup-image /StartComponentCleanup /Resetbase" -Wait -LoadUserProfile
```

Example:

PowerShell

```
Start-Process "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Deployment Tools\amd64\DISM\dism.exe" -ArgumentList " /Image:`"C:\Mount`" /Cleanup-image /StartComponentCleanup /Resetbase /Defer" -Wait -LoadUserProfile

Start-Process "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Deployment Tools\amd64\DISM\dism.exe" -ArgumentList " /Image:"C:\Mount" /Cleanup-image /StartComponentCleanup /Resetbase" -Wait -LoadUserProfile
```

For more information, see Modify a Windows image using DISM: Reduce the size of an image, DISM Operating System Package (.cab or .msu) Servicing Command-Line Options: /Cleanup-Image, and Start-Process.

Step 10: Verify all desired packages have been added to boot image

After the optional components and the cumulative update (CU) have been applied to the boot image, verify that they're showing as installed:

PowerShell

From an elevated **PowerShell** command prompt, run the following command to verify that all optional components and the cumulative update (CU) have been applied to the boot image:

PowerShell

```
Get-WindowsPackage -Path "<Mount_folder_path>"
```

Example:

PowerShell

```
Get-WindowsPackage -Path "C:\Mount"
```

For more information, see Get-WindowsPackage.

Step 11: Unmount boot image and save changes

Once drivers, optional components, and the cumulative update (CU) have been applied to the boot image, unmount the boot image and save changes.

PowerShell

From an elevated **PowerShell** command prompt, run the following command to unmount the boot image and save changes:

PowerShell

```
Dismount-WindowsImage -Path "<Mount_folder_path>" -Save -Verbose
```

Example:

PowerShell

```
Dismount-WindowsImage -Path "C:\Mount" -Save -Verbose
```

For more information, see Dismount-WindowsImage.

Step 12: Export boot image to reduce size

1. Once the boot image has been unmounted and saved, its size can be further reduced by exporting it:

   PowerShell

   From an elevated **PowerShell** command prompt, run the following command to further reduce the size of the boot image by exporting it:

   PowerShell

   ```
   Export-WindowsImage -SourceImagePath "<Boot_image_path>\<boot_image>.wim" -
   SourceIndex 1 -DestinationImagePath "<Boot_image_path>\<boot_image>-
   export.wim" -CompressionType max -Verbose
   ```

   Example:

PowerShell

```
Export-WindowsImage -SourceImagePath "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environment\amd64\en-us\winpe.wim" -SourceIndex 1 -DestinationImagePath "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environment\amd64\en-us\winpe-export.wim" -CompressionType max -Verbose
```

For more information, see Export-WindowsImage.

2. Once the export has completed:

   a. Delete the original updated boot image:

   PowerShell

   From an elevated **PowerShell** command prompt, run the following command to delete the original updated boot image:

   PowerShell

   ```
   Remove-Item -Path "<Boot_image_path>\<boot_image>.wim" -Force
   ```

   Example:

   PowerShell

   ```
   Remove-Item -Path "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environment\amd64\en-us\winpe.wim" -Force
   ```

   For more information, see Remove-Item.

   b. Rename the exported boot image with the name of the original boot image:

   PowerShell

   From an elevated **PowerShell** command prompt, run the following command to rename the exported boot image with the name of the original boot image:

   PowerShell

   ```
   Rename-Item -Path "<Boot_image_path>\<exported_boot_image>.wim" -NewName "<original_boot_image_name>.wim"
   ```

Example:

PowerShell

```
Rename-Item -Path "C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environment\amd64\en-us\winpe-export.wim" -NewName "winpe.wim"
```

For more information, see Rename-Item.

Step 13: Update boot image in products that utilize it (if applicable)

After the default `winpe.wim` boot image from the Windows ADK has been updated, additional steps usually need to take place in the products that utilize the boot image. The following links contain information on how to update the boot image for several popular products that utilize boot images:

- Microsoft Configuration Manager
- Microsoft Deployment Toolkit (MDT)
- Windows Deployment Services
  - Original WDS boot image is updated
  - WDS boot image is replaced with new updated boot image
  - Add updated boot image as a new boot image in WDS

For any other products that utilize boot images, consult the product's documentation on updating the boot image.

Microsoft Configuration Manager considerations

How Microsoft Configuration Manager creates boot images

Microsoft Configuration Manager creates its own boot images by taking the `winpe.wim` from the Windows ADK, adding some optional components it requires to function correctly, and then saving the boot image as `boot.wim` in the directory `<ConfigMgr_Install_Directory>\OSD\boot\<architecture>\boot.wim`. This `boot.wim` boot image is considered the pristine authoritative copy of the boot image by Configuration Manager. Configuration Manager never touches, modifies, or updates the `boot.wim` boot image except in some specific scenarios. Instead, when changes are done in the properties of the boot image in Configuration Manager such as:

- Adding drivers
- Adding optional components
- Enabling the command prompt

Configuration Manager makes a copy of `boot.wim`, applies the changes to the copy, and then saves the new boot image as `boot.<package_id>.wim`.

If in the future any additional changes are done to the boot image, Configuration Manager discards the previously created `boot.<package_id>.wim` boot image, makes a new copy of `boot.wim`, applies the changes to the copy, and then saves the new boot image as `boot.<package_id>.wim`. In other words, `boot.wim` is never touched. Anytime any changes are made to a boot image, both the new changes and any changes done in the past are all reapplied to a new copy of `boot.wim`.

This process has the following advantages:

1. Keeps `boot.wim` pristine.

2. Makes sure that changes done to a boot image are being done to a pristine unmodified version of the boot image. This process helps avoid corruption when a boot image is updated multiple times. I can also correct issues with existing boot images.

3. Helps manage components in the boot image. The process doesn't need to know what components may need to be removed from the boot image each time the boot image is rebuilt. Instead, it just needs to know what components need to be added to the boot image.

4. It reduces the size of the boot image which can occur when components are repeatedly added to and removed from the boot image.

Configuration Manager updates the `boot.wim` boot image in two scenarios:

1. When Configuration Manager is upgraded between versions or a hotfix roll-up (HFRU) is applied, `boot.wim` may be updated as part of the upgrade process.

2. When selecting the option **Reload this boot image with the current Windows PE version from the Windows ADK** in the **Update Distribution Points Wizard**.

In these scenarios, the `boot.wim` boot image is updated using the `winpe.wim` boot image from the Windows ADK as described earlier in this section. This process creates a new pristine copy of the `boot.wim` boot image using the current version of the `winpe.wim` boot image that is part of the Windows ADK.

Which boot image should be updated with the cumulative update?

When manually adding a cumulative update to a Configuration Manager boot image, it's recommended to update the `winpe.wim` boot image from the Windows ADK instead of directly updating the `boot.wim` boot image generated by Configuration Manager.

The `winpe.wim` boot image from the Windows ADK should be updated because if `boot.wim` generated by Configuration Manager is updated instead, then the next time `boot.wim` is updated via a Configuration Manager upgrade or the **Reload this boot image with the current Windows PE version from the Windows ADK** option, then changes made to `boot.wim`, including the applied cumulative update, will be lost. If the `winpe.wim` boot image from the Windows ADK is updated with the cumulative update instead, then the cumulative update persists and is preserved even when Configuration Manager does update the `boot.wim` boot image.

> **Important**
>
> Never manually update the `boot.<package_id>.wim` boot image. In addition to facing the same issues when manually updating the `boot.wim` boot image, the `boot.<package_id>.wim` boot image will also face additional issues such as:
>
> - Any time any changes are done to the boot image (adding drivers, enabling the command prompt, etc.), any manual changes done to the boot image, including the cumulative update, will be lost.
> - Manually changing the `boot.<package_id>.wim` boot image changes the hash value of the boot image. A change in the hash value of the boot image can lead to download failures when downloading the boot image from a distribution point.

Updating `winpe.wim` from the Windows ADK ensures that the cumulative update stays applied regardless of what changes are made to the `boot.wim` boot image via Configuration Manager.

Add optional components manually to Configuration Manager boot images

For Microsoft Configuration Manager boot images, when applying a cumulative update to a boot image, make sure to add any desired optional components manually using the command lines from the walkthrough instead of adding them through Configuration Manager. Optional

components are added to boot images in Configuration Manager via the **Optional Components** tab in the **Properties** of the boot image.

Optional components need to be added to the boot image manually instead of via Configuration Manager because:

- When the cumulative update is applied, it also updates any optional components as needed.
- If optional components are added through Configuration Manager on a boot image that has a cumulative update, then the optional components aren't updated with the cumulative update. Adding the optional components through Configuration Manager could lead to unexpected behaviors and problems. The cumulative update needs to be added after the optional components have been added to the boot image for the optional components to be updated properly with the cumulative update.

> **Note**
>
> If an optional component is attempted to be added via the **Optional Components** tab in the **Properties** of the boot image in Configuration Manager but the optional component has already been manually added to the boot image, Configuration Manager won't add that optional component again. Instead, Configuration Manager detects that the optional component has already been added and it won't try to add the optional component again.

Configuration Manager boot image required components

For Microsoft Configuration Manager boot images to function correctly, it requires the following optional components:

Expand table

Feature	File Name	Dependency	Required by ConfigMgr
Scripting/WinPE-Scripting	`WinPE-Scripting.cab`	NA	Yes
Scripting/WinPE-WMI	`WinPE-WMI.cab`	NA	Yes
Network/WinPE-WDS-Tools	`WinPE-WDS-Tools.cab`	NA	Yes
Startup/WinPE-SecureStartup	`WinPE-SecureStartup.cab`	Scripting/WinPE-WMI	Yes

When adding optional components to any boot image used by Configuration Manager during the Step 6: Add optional components to boot image step, make sure to first add the above required components in the above order to the boot image. After adding the required components to the boot image, add any additional desired optional components to the boot image.

For a list of all available WinPE optional components including descriptions for each component, see WinPE Optional Components (OC) Reference: WinPE Optional Components.

Updating the boot image in Configuration Manager

After updating the `winpe.wim` boot image from the Windows ADK, generate a new `boot.wim` boot image for Configuration Manager so that it contains the cumulative update. A new `boot.wim` boot image can be generated by using the following steps:

1. Open the Microsoft Configuration Manager console.
2. In the Microsoft Configuration Manager console, navigate to **Software Library** > **Overview** > **Operating Systems** > **Boot Images**.
3. In the **Boot Images** pane, select the desired boot image.
4. In the toolbar, select **Update Distribution Points**.
5. When the **Update Distribution Points Wizard** window that appears:
   a. In the **General/Update distribution points with this image** page, select the **Reload this boot image with the current Windows PE version from the Windows ADK** option, and then select the **Next >** button.
   b. In the **Summary** page, select the **Next >** button.
   c. The **Progress** page appears while the boot image builds.
   d. Once the boot image finishes building, the **The task "Update Distribution Points Wizard" completed successfully/Completion** page appears. Select the **Close** button.

This process updates the boot image used by Configuration Manager. It also updates the boot image and the bootmgr boot files used by any PXE-enabled distribution points.

> **ⓘ Important**
>
> If there are multiple boot images used in the environment for PXE-enabled distribution points, make sure to update all of the PXE-enabled boot images with the same cumulative

update. This will ensure that the PXE-enabled distribution points all use the version of the bootmgr boot files extracted from the boot images (if applicable).

Updating Configuration Manager boot media

After completing the walkthrough, including updating boot images in Configuration Manager, update any Configuration Manager task sequence media. Updating any Configuration Manager task sequence media ensures that the task sequence media has both the updated boot image. If applicable, it will also update bootmgr boot files on the media by extracting the latest versions from the boot image. For more information on creating Configuration Manager task sequence media, see Create task sequence media.

Microsoft Deployment Toolkit (MDT) considerations

When adding a cumulative update to a Microsoft Deployment Toolkit (MDT) boot image, it's recommended to update the `winpe.wim` boot image from the Windows ADK instead of directly updating the `LiteTouchPE_<arch>.wim` boot image in the MDT Deployment Share.

The `winpe.wim` boot image from the Windows ADK should be updated because if `LiteTouchPE_<arch>.wim` is updated instead, then the next time the MDT Deployment Share is updated, the changes made to `LiteTouchPE_<arch>.wim`, including the applied cumulative update, will be lost. If the `winpe.wim` boot image from the Windows ADK is updated with the cumulative update instead, then the cumulative update persists and is preserved even when the MDT Deployment Share is updated.

MDT and Windows ADK versions

Microsoft Deployment Toolkit (MDT) doesn't support versions of Windows or the Windows ADK beyond Windows 10. When MDT is used, the recommendation is to use the ADK for Windows 10, version 2004 instead of the latest version of the Windows ADK. **ADK for Windows 10, version 2004** was the last version of the Windows ADK supported by MDT. When updating the boot image for the ADK for Windows 10, version 2004 with a cumulative update, use the cumulative update for Windows 10 Version 22H2.

MDT boot image required components

For Microsoft Deployment Toolkit (MDT) boot images to function correctly, it requires the following optional components:

Expand table

Feature	File Name	Dependency	Required by MDT
Scripting/WinPE-Scripting	`WinPE-Scripting.cab`	NA	Yes
Scripting/WinPE-WMI	`WinPE-WMI.cab`	NA	Yes
File management/WinPE-FMAPI	`WinPE-FMAPI.cab`	NA	Yes
Startup/WinPE-SecureStartup	`WinPE-SecureStartup.cab`	Scripting/WinPE-WMI	Yes
HTML/WinPE-HTA	`WinPE-HTA.cab`	Scripting/WinPE-WMI	Yes

When adding optional components to any boot image used by MDT during the Step 6: Add optional components to boot image step, make sure to first add the above required components in the above order to the boot image. After adding the required components to the boot image, add any additional desired optional components to the boot image.

For a list of all available WinPE optional components including descriptions for each component, see WinPE Optional Components (OC) Reference: WinPE Optional Components.

Updating the boot image and boot media in MDT

After updating the `winpe.wim` boot image from the Windows ADK, generate a new `LiteTouchPE_<arch>.wim` boot image for MDT that contains the cumulative update followed by creating new MDT boot media. New MDT boot images and MDT boot media can be generated by using the following steps:

1. Make sure Step 8: Copy boot files from mounted boot image to ADK installation path has been completed. MDT copies the bootmgr boot files from the Windows ADK installation path to its deployment share. Following this step makes sure that the deployment share has the latest bootmgr boot files that are needed when creating MDT boot media.

2. Open the Microsoft Deployment Toolkit (MDT) Deployment Workbench console.

3. In the Deployment Workbench console, navigate to **Deployment Workbench** > **Deployment Shares** > **MDT Deployment Share**.

4. Right click on **MDT Deployment Share** and select **Update Deployment Share**.

5. In the **Update Deployment Share Wizard** window that appears:

    a. In the **Options** page, select the **Completely regenerate the boot images** option, and then select the **Next** > button.

b. In the **Summary** page, select the **Next** > button.

c. The **Progress** page appears while the boot image and deployment share build.

d. Once the boot image and deployment share finish building, the **The process completed successfully/Confirmation** page appears. Select the **Finish** button.

These steps also update the MDT boot media in the MDT Deployment Share. After following the above steps, use the newly updated ISO files in the `<DeploymentShare>\Boot` folder to create new MDT boot media.

Windows Deployment Services (WDS) considerations

Original WDS boot image is updated

If the WDS boot image modified was the original WDS boot image in the `<RemoteInstall>` folder, then the only additional step to take is to restart `Windows Deployment Services Server` service. WDS can be restarted by using the following command lines:

PowerShell

From an elevated **PowerShell** command prompt, run the following command to restart the `Windows Deployment Services Server` service:

PowerShell

```
Restart-Service -Name WDSServer
```

For more information, see Restart-Service.

WDS boot image is replaced with new updated boot image

In the following boot image replacement scenario for WDS:

- The boot image modified as part of this guide is outside of the `<RemoteInstall>` folder. For example, the `winpe.wim` boot image that comes with the Windows ADK.
- An existing boot image in WDS is being replaced with the updated boot image.

Follow these steps to update the boot image in WDS:

1. Replace the existing boot image in WDS with the modified boot image using the following command lines:

   PowerShell

   In PowerShell, the original boot image needs to be removed first and then replaced with a new image. From an elevated **PowerShell** command prompt, run the following commands to replace an existing boot image in WDS with a new boot image:

   PowerShell

   ```
   Remove-WdsBootImage -Architecture <Architecture_x64_or_x86> -ImageName "
   <Name_Of_Existing_Boot_Image_In_WDS>"
   Import-WdsBootImage -Path "<Path_To_Updated_Boot_Image>\<boot_image>.wim" -
   NewImageName "<Name_Of_Existing_Boot_Image_In_WDS>"
   ```

   Example:

   PowerShell

   ```
   Remove-WdsBootImage -Architecture x64 -ImageName "Microsoft Windows PE
   (amd64)"
   Import-WdsBootImage -Path "C:\Program Files (x86)\Windows
   Kits\10\Assessment and Deployment Kit\Windows Preinstallation
   Environment\amd64\en-us\winpe.wim" -NewImageName "Microsoft Windows PE
   (amd64)"
   ```

   For more information, see Remove-WdsBootImage and Import-WdsBootImage.

2. Once the existing boot image in WDS has been replaced, restart the WDS service:

   PowerShell

   From an elevated **PowerShell** command prompt, run the following command to restart the `Windows Deployment Services Server` service:

   PowerShell

   ```
   Restart-Service -Name WDSServer
   ```

   For more information, see Restart-Service.

Add updated boot image as a new boot image in WDS

In the following boot image scenario for WDS:

- The boot image modified as part of this guide is outside of the `<RemoteInstall>` folder. For example, the `winpe.wim` boot image that comes with the Windows ADK
- The updated boot image is being added as a new boot image in WDS

Follow these steps to add the boot image in WDS:

1. Add the updated boot image to WDS using the following command lines:

   PowerShell

   From an elevated **PowerShell** command prompt, run the following commands to add the updated boot image in WDS as a new boot image:

   PowerShell
   ```
   Import-WdsBootImage -Path "<Path_To_Updated_Boot_Image>\<boot_image>.wim" -
   NewImageName "<Boot_Image_Name_In_WDS>"
   ```

   Example:

   PowerShell
   ```
   Import-WdsBootImage -Path "C:\Program Files (x86)\Windows
   Kits\10\Assessment and Deployment Kit\Windows Preinstallation
   Environment\amd64\en-us\winpe.wim" -NewImageName "Microsoft Windows PE
   (amd64) - Updated"
   ```

   For more information, see Import-WdsBootImage.

2. Once the existing boot image in WDS has been replaced, restart the WDS service:

   PowerShell

   From an elevated **PowerShell** command prompt, run the following command to restart the `Windows Deployment Services Server` service:

   PowerShell
   ```
   Restart-Service -Name WDSServer
   ```

   For more information, see Restart-Service.

Boot.wim support

The **boot.wim** that is part of Windows installation media isn't supported for deploying Windows 11 with Windows Deployment Services (WDS). Additionally, the **boot.wim** from Windows 11 installation media isn't supported for deploying any version of Windows with Windows Deployment Services (WDS). For more information, see Windows Deployment Services (WDS) boot.wim support.

Windows Server 2012 R2

This walk-through isn't intended for use with Windows Server 2012 R2. The steps in this article may work with Windows Server 2012 R2 when using older versions of the Windows ADK. However, it may have compatibility problems with versions of the Windows ADK that are newer than the ADK for Windows 10, version 2004. To resolve compatibility problems with newer ADKs and Windows Server 2012 R2:

1. Upgrade Windows Server 2012 R2 to a newer version of Windows Server.
2. Perform the boot image customizations on a computer running a version of Windows that supports the newer ADKs, for example Windows 10 or Windows 11, and then transfer the modified boot image to the Windows Server 2012 R2 server.

For more information, see Windows Server 2012 R2 Lifecycle.

Related articles

- Create bootable Windows PE media: Update the Windows PE add-on for the Windows ADK
- Update Windows installation media with Dynamic Update: Update WinPE
- KB5025885: How to manage the Windows Boot Manager revocations for Secure Boot changes associated with CVE-2023-24932: Updating bootable media

---

Last updated on 08/15/2025

MBR2GPT.EXE

Applies to: ✅ Windows 11, ✅ Windows 10

> **ⓘ Important**
>
> **MBR2GPT.EXE** is located in the `Windows\System32` directory on any device running a currently supported version of Windows.

MBR2GPT.EXE converts a disk from the Master Boot Record (MBR) to the GUID Partition Table (GPT) partition style without modifying or deleting data on the disk. The tool runs from a Windows Preinstallation Environment (Windows PE) command prompt, but can also be run from the full Windows operating system (OS) by using the `/allowFullOS` option.

The tool is available in both the full OS environment and Windows PE.

See the following video for a detailed description and demonstration of MBR2GPT.

https://www.youtube-nocookie.com/embed/hfJep4hmg9o

MBR2GPT can be used to:

- Convert any attached MBR-formatted system disk to the GPT partition format. The tool can't be used to convert non-system disks from MBR to GPT.
- Convert an MBR disk with BitLocker-encrypted volumes as long as protection is suspended. To resume BitLocker after conversion, the existing protectors need to be deleted and then recreated.
- Convert an operating system disk from MBR to GPT using Microsoft Configuration Manager or Microsoft Deployment Toolkit (MDT).

Offline conversion of system disks with earlier versions of Windows installed, such as Windows 7, 8, or 8.1 aren't officially supported. The recommended method to convert these disks is to upgrade the operating system to a currently supported version of Windows, then perform the MBR to GPT conversion.

> **ⓘ Important**
>
> After the disk has been converted to GPT partition style, the firmware must be reconfigured to boot in UEFI mode.
>
> Make sure the device supports UEFI before attempting to convert the disk.

Disk Prerequisites

Before any change to the disk is made, MBR2GPT validates the layout and geometry of the selected disk to ensure that:

- The disk is currently using MBR
- There's enough space not occupied by partitions to store the primary and secondary GPTs:
  - 16 KB + 2 sectors at the front of the disk
  - 16 KB + 1 sector at the end of the disk
- There are at most three primary partitions in the MBR partition table
- One of the partitions is set as active and is the system partition
- The disk doesn't have any extended/logical partition
- The BCD store on the system partition contains a default OS entry pointing to an OS partition
- The volume IDs can be retrieved for each volume that has a drive letter assigned
- All partitions on the disk are of MBR types recognized by Windows or has a mapping specified using the `/map` command-line option

If any of these checks fails, the conversion doesn't proceed, and an error is returned.

Syntax

```
MBR2GPT /validate|convert [/disk:<diskNumber>] [/logs:<logDirectory>] [/map:<source>=<destination>] [/allowFullOS]
```

Options

Expand table

Option	Description
/validate	Instructs `MBR2GPT.exe` to perform only the disk validation steps and report whether the disk is eligible for conversion.
/convert	Instructs `MBR2GPT.exe` to perform the disk validation and to proceed with the conversion if all validation tests pass.
/disk: ***<diskNumber>***	Specifies the disk number of the disk to be converted to GPT. If not specified, the system disk is used. The mechanism used is the same as used by the diskpart.exe tool **SELECT DISK SYSTEM** command.

Option	Description
/logs: *<logDirectory>*	Specifies the directory where `MBR2GPT.exe` logs should be written. If not specified, **%windir%** is used. If specified, the directory must already exist, it isn't automatically created or overwritten.
/map:*<source>***=** *<destination>*	Specifies other partition type mappings between MBR and GPT. The MBR partition number is specified in decimal notation, not hexadecimal. The GPT GUID can contain brackets, for example: **/map:42={af9b60a0-1431-4f62-bc68-3311714a69ad}**. Multiple /map options can be specified if multiple mappings are required.
/allowFullOS	By default, `MBR2GPT.exe` can only run from Windows PE and is blocked from running in full Windows. This option overrides this block and enables disk conversion while running in the full Windows environment. **Note**: Since the existing MBR system partition is in use while running the full Windows environment, it can't be reused. In this case, a new EFI system partition is created by shrinking the OS partition.

Examples

Validation example

In the following example, disk 0 is validated for conversion. Errors and warnings are logged to the default location of `%windir%`.

Windows Command Prompt

```
X:\> mbr2gpt.exe /validate /disk:0
MBR2GPT: Attempting to validate disk 0
MBR2GPT: Retrieving layout of disk
MBR2GPT: Validating layout, disk sector size is: 512
MBR2GPT: Validation completed successfully
```

Conversion example

In the following example:

1. Using DiskPart the current disk partition layout is displayed before the conversion. Three partitions are present on the MBR disk (disk 0):

    - A system reserved partition.
    - A Windows partition.
    - A recovery partition.
    - A DVD-ROM is also present as volume 0.

2. The OS volume is selected, partitions are listed, and partition details are displayed for the OS partition. The MBR partition type is **07** corresponding to the installable file system (IFS) type.

3. The MBR2GPT tool is used to convert disk 0.

4. The DiskPart tool displays that disk 0 is now using the GPT format.

5. The new disk layout is displayed - four partitions are present on the GPT disk: three are identical to the previous partitions and one is the new EFI system partition (volume 3).

6. The OS volume is selected again. The detail displays that the OS volume is converted to the GPT partition type of **ebd0a0a2-b9e5-4433-87c0-68b6b72699c7** corresponding to the **PARTITION_BASIC_DATA_GUID** type.

As noted in the output from the MBR2GPT tool, changes to the computer firmware need to be made so that the new EFI system partition boots properly.

▼ Expand to show MBR2GPT example

Windows Command Prompt

```
X:\>DiskPart.exe

Microsoft DiskPart version 10.0.15048.0

Copyright (C) Microsoft Corporation.
On computer: MININT-K71F13N

DISKPART> list volume

  Volume ###  Ltr  Label        Fs     Type        Size     Status     Info
  ----------  ---  -----------  -----  ----------  -------  ---------  --------
  Volume 0     F   CENA_X64FRE  UDF    DVD-ROM     4027 MB  Healthy
  Volume 1     C   System Rese  NTFS   Partition    499 MB  Healthy
  Volume 2     D   Windows      NTFS   Partition     58 GB  Healthy
  Volume 3     E   Recovery     NTFS   Partition    612 MB  Healthy    Hidden

DISKPART> select volume 2

Volume 2 is the selected volume.

DISKPART> list partition

  Partition ###  Type              Size     Offset
  -------------  ----------------  -------  -------
  Partition 1    Primary            499 MB  1024 KB
* Partition 2    Primary             58 GB   500 MB
  Partition 3    Recovery           612 MB    59 GB
```

```
DISKPART> detail partition

Partition 2
Type  : 07
Hidden: No
Active: No
Offset in Bytes: 524288000

  Volume ###  Ltr  Label        Fs     Type        Size     Status     Info
  ----------  ---  -----------  -----  ----------  -------  ---------  --------
* Volume 2     D   Windows      NTFS   Partition     58 GB  Healthy

DISKPART> exit

Leaving DiskPart...

X:\>mbr2gpt /convert /disk:0

MBR2GPT will now attempt to convert disk 0.
If conversion is successful the disk can only be booted in GPT mode.
These changes cannot be undone!

MBR2GPT: Attempting to convert disk 0
MBR2GPT: Retrieving layout of disk
MBR2GPT: Validating layout, disk sector size is: 512 bytes
MBR2GPT: Trying to shrink the system partition
MBR2GPT: Trying to shrink the OS partition
MBR2GPT: Creating the EFI system partition
MBR2GPT: Installing the new boot files
MBR2GPT: Performing the layout conversion
MBR2GPT: Migrating default boot entry
MBR2GPT: Adding recovery boot entry
MBR2GPT: Fixing drive letter mapping
MBR2GPT: Conversion completed successfully
MBR2GPT: Before the new system can boot properly you need to switch the firmware to
boot to UEFI mode!

X:\>DiskPart

Microsoft DiskPart version 10.0.15048.0

Copyright (C) Microsoft Corporation.
On computer: MININT-K71F13N

DISKPART> list disk

  Disk ###  Status         Size     Free     Dyn  Gpt
  --------  -------------  -------  -------  ---  ---
  Disk 0    Online           60 GB      0 B        *

DISKPART> select disk 0

Disk 0 is now the selected disk.

DISKPART> list volume
```

```
  Volume ###  Ltr  Label        Fs     Type        Size     Status     Info
  ----------  ---  -----------  -----  ----------  -------  ---------  --------
  Volume 0     F   CENA_X64FRE  UDF    DVD-ROM     4027 MB  Healthy
  Volume 1     D   Windows      NTFS   Partition     58 GB  Healthy
  Volume 2     C   System Rese  NTFS   Partition    499 MB  Healthy    Hidden
  Volume 3                      FAT32  Partition    100 MB  Healthy    Hidden
  Volume 4     E   Recovery     NTFS   Partition    612 MB  Healthy    Hidden

DISKPART> select volume 1

Volume 1 is the selected volume.

DISKPART> list partition

  Partition ###  Type              Size     Offset
  -------------  ----------------  -------  -------
  Partition 1    Recovery           499 MB  1024 KB
* Partition 2    Primary             58 GB   500 MB
  Partition 4    System             100 MB    59 GB
  Partition 3    Recovery           612 MB    59 GB

DISKPART> detail partition

Partition 2
Type    : ebd0a0a2-b9e5-4433-87c0-68b6b72699c7
Hidden  : No
Required: No
Attrib  : 0000000000000000
Offset in Bytes: 524288000

  Volume ###  Ltr  Label        Fs     Type        Size     Status     Info
  ----------  ---  -----------  -----  ----------  -------  ---------  --------
* Volume 1     D   Windows      NTFS   Partition     58 GB  Healthy
```

Specifications

Disk conversion workflow

The following steps illustrate high-level phases of the MBR-to-GPT conversion process:

1. Disk validation is performed.
2. The disk is repartitioned to create an EFI system partition if one doesn't already exist.
3. UEFI boot files are installed to the EFI system partition.
4. GPT metadata and layout information are applied.
5. The boot configuration data (BCD) store is updated.
6. Drive letter assignments are restored.

Creating an EFI system partition

For Windows to remain bootable after the conversion, an EFI system partition must be in place. MBR2GPT creates the EFI system partition using the following rules:

1. The existing MBR system partition is reused if it meets these requirements:
    - It isn't also the OS or Windows Recovery Environment partition.
    - It is at least 100 MB (or 260 MB for 4K sector size disks) in size.
    - It's less than or equal to 1 GB in size. This size is a safety precaution to ensure it isn't a data partition.
    - The conversion isn't being performed from the full OS. In this case, the existing MBR system partition is in use and can't be repurposed.
2. If the existing MBR system partition can't be reused, a new EFI system partition is created by shrinking the OS partition. This new partition has a size of 100 MB (or 260 MB for 4K sector size disks) and is formatted FAT32.

If the existing MBR system partition isn't reused for the EFI system partition, it's no longer used by the boot process after the conversion. Other partitions aren't modified.

> **Important**
>
> If the existing MBR system partition isn't reused for the EFI system partition, it might be assigned a drive letter. If this small partition isn't going to be used, its drive letter must be manually hidden.

Partition type mapping and partition attributes

Since GPT partitions use a different set of type IDs than MBR partitions, each partition on the converted disk must be assigned a new type ID. The partition type mapping follows these rules:

1. The EFI system partition is always set to partition type **PARTITION_SYSTEM_GUID** (**c12a7328-f81f-11d2-ba4b-00a0c93ec93b**).
2. If an MBR partition is of a type that matches one of the entries specified in the `/map` switch, the specified GPT partition type ID is used.
3. If the MBR partition is of type **0x27**, the partition is converted to a GPT partition of type **PARTITION_MSFT_RECOVERY_GUID** (**de94bba4-06d1-4d40-a16a-bfd50179d6ac**).
4. All other MBR partitions recognized by Windows are converted to GPT partitions of type **PARTITION_BASIC_DATA_GUID** (**ebd0a0a2-b9e5-4433-87c0-68b6b72699c7**).

In addition to applying the correct partition types, partitions of type PARTITION_MSFT_RECOVERY_GUID also have the following GPT attributes set:

- **GPT_ATTRIBUTE_PLATFORM_REQUIRED** (**0x0000000000000001**)
- **GPT_BASIC_DATA_ATTRIBUTE_NO_DRIVE_LETTER** (**0x8000000000000000**)

For more information about partition types, see:

- GPT partition types
- MBR partition types

Persisting drive letter assignments

The conversion tool attempts to remap all drive letter assignment information contained in the registry that corresponds to the volumes of the converted disk. If a drive letter assignment can't be restored, an error is displayed at the console and in the log, so that correct assignment of the drive letter can be manually performed.

> ⓘ **Important**
>
> This code runs after the layout conversion takes place, so the operation can't be undone at this stage.

The conversion tool will obtain volume unique ID data before and after the layout conversion, organizing this information into a lookup table. It then iterates through all the entries in **HKLM\SYSTEM\MountedDevices**, and for each entry it does the following:

1. Checks if the unique ID corresponds to any of the unique IDs for any of the volumes that are part of the converted disk.
2. If found, set the value to be the new unique ID, obtained after the layout conversion.
3. If the new unique ID can't be set and the value name starts with **\DosDevices**, issue a console and log warning about the need for manual intervention in properly restoring the drive letter assignment.

Troubleshooting

The tool displays status information in its output. Both validation and conversion are clear if any errors are encountered. For example, if one or more partitions don't translate properly, this information is displayed and the conversion not performed. To view more detail about any errors that are encountered, see the associated log files.

Logs

Four log files are created by the MBR2GPT tool:

- diagerr.xml
- diagwrn.xml
- setupact.log
- setuperr.log

These files contain errors and warnings encountered during disk validation and conversion. Information in these files can be helpful in diagnosing problems with the tool. The `setupact.log` and `setuperr.log` files have the most detailed information about disk layouts, processes, and other information pertaining to disk validation and conversion.

> **Note**
>
> The *setupact.log** files are different than the Windows Setup files that are found in the `%Windir%\Panther` directory.

The default location for all these log files in Windows PE is **%windir%**.

Interactive help

To view a list of options available when using the tool, enter the following command in an elevated command prompt:

Windows Command Prompt

```
mbr2gpt.exe /?
```

The following text is displayed:

Windows Command Prompt

```
C:\> mbr2gpt.exe /?

Converts a disk from MBR to GPT partitioning without modifying or deleting data on
the disk.

MBR2GPT.exe /validate|convert [/disk:<diskNumber>] [/logs:<logDirectory>] [/map:
<source>=<destination>] [/allowFullOS]

Where:

 /validate
```

```
        - Validates that the selected disk can be converted
          without performing the actual conversion.

/convert
        - Validates that the selected disk can be converted
          and performs the actual conversion.

/disk:<diskNumber>
        - Specifies the disk number of the disk to be processed.
          If not specified, the system disk is processed.

/logs:<logDirectory>
        - Specifies the directory for logging. By default logs
          are created in the %windir% directory.

/map:<source>=<destination>
        - Specifies the GPT partition type to be used for a
          given MBR partition type not recognized by Windows.
          Multiple /map switches are allowed.

/allowFullOS
        - Allows the tool to be used from the full Windows
          environment. By default, this tool can only be used
          from the Windows Preinstallation Environment.
```

Return codes

MBR2GPT has the following associated return codes:

Expand table

Return code	Description
0	Conversion completed successfully.
1	Conversion was canceled by the user.
2	Conversion failed due to an internal error.
3	Conversion failed due to an initialization error.
4	Conversion failed due to invalid command-line parameters.
5	Conversion failed due to error reading the geometry and layout of the selected disk.
6	Conversion failed because one or more volumes on the disk is encrypted.
7	Conversion failed because the geometry and layout of the selected disk don't meet requirements.

Return code	Description
8	Conversion failed due to error while creating the EFI system partition.
9	Conversion failed due to error installing boot files.
10	Conversion failed due to error while applying GPT layout.
100	Conversion to GPT layout succeeded, but some boot configuration data entries couldn't be restored.

Determining the partition type

The partition type can be determined in one of three ways:

- Using Windows PowerShell
- Using the Disk Management tool
- Using the DiskPart tool

Windows PowerShell

The following command can be entered at a Windows PowerShell prompt to display the disk number and partition type:

PowerShell

```
Get-Disk | ft -Auto
```

Example output:

PowerShell

```
PS C:\> Get-Disk | ft -Auto

Number Friendly Name      Serial Number        HealthStatus OperationalStatus Total
Size Partition Style
------ -------------      -------------        ------------ ----------------- ------
---- ---------------
0      MTFDDAK256MAM-1K1  13050928F47C         Healthy      Online
238.47 GB MBR
1      ST1000DM003-1ER162 Z4Y3GD8F             Healthy      Online
931.51 GB GPT
```

Disk Management tool

The partition type of a disk can be viewed by using the Disk Management tool:

1. Right-click on the Start Menu and select **Disk Management**. Alternatively, right-click on the Start Menu and select **Run**. In the **Run** dialog box that appears, enter `diskmgmt.msc` and then select **OK**.

2. In the **Disk Management** window that appears:

   a. On the bottom pane, select the disk number of interest.

   b. Select the **Action** menu and then select **All Tasks** > **Properties**. Alternatively, right-click on the disk number of interest and select **Properties**.

   c. In the **Properties** dialog box that appears for the disk, select the **Volumes** tab.

   d. Under the **Volumes** tab, the partition type is displayed next to **Partition style:**.

DiskPart tool

The partition type can be determined with the DiskPart tool. The DiskPart tool is useful in scenarios where the Disk Management tool and PowerShell aren't available, such as in WinPE. PowerShell isn't available in WinPE when the PowerShell optional component isn't loaded. To use the DiskPart tool to determine the partition type:

1. Open an elevated command prompt.

2. In the elevated command prompt that opens enter the following command:

   Windows Command Prompt
   ```
   DiskPart.exe
   ```

3. The **DISKPART>** prompt is displayed in the command prompt windows. At the **DISKPART>** prompt, enter the following command:

   Windows Command Prompt
   ```
   list disk
   ```

4. The partition type is displayed in the **Gpt** column. If the partition is GPT, an asterisk (*) is displayed in the column. If the partition is MBR, the column is blank.

The following shows an example output of the DiskPart tool showing the partition type for two disks:

Windows Command Prompt

```
X:\>DiskPart.exe

Microsoft DiskPart version 10.0.15048.0

Copyright (C) Microsoft Corporation.
On computer: MININT-K71F13N

DISKPART> list disk

  Disk ###  Status         Size     Free     Dyn  Gpt
  --------  -------------  -------  -------  ---  ---
  Disk 0    Online          238 GB      0 B
  Disk 1    Online          931 GB      0 B        *
```

In this example, Disk 0 is formatted with the MBR partition style, and Disk 1 is formatted using GPT.

Last updated on 05/07/2025

Configure a PXE server to load Windows PE

Article • 02/28/2023

Applies to:

- Windows 10

This walkthrough describes how to configure a PXE server to load Windows PE by booting a client computer from the network. Using the Windows PE tools and a Windows 10 image file, you can install Windows 10 from the network.

Prerequisites

- A deployment computer: A computer with the Windows Assessment and Deployment Kit (Windows ADK) and the Windows PE add-on with ADK installed.
- A DHCP server: A DHCP server or DHCP proxy configured to respond to PXE client requests is required.
- A PXE server: A server running the TFTP service that can host Windows PE boot files that the client will download.
- A file server: A server hosting a network file share.

All four of the roles specified above can be hosted on the same computer or each can be on a separate computer.

Step 1: Copy Windows PE source files

1. On the deployment computer, select **Start**, and type **deployment**.

2. Right-click **Deployment and Imaging Tools Environment** and then select **Run as administrator**. The Deployment and Imaging Tools Environment shortcut opens a Command Prompt window and automatically sets environment variables to point to all the necessary tools.

3. Run the following command to copy the base Windows PE files into a new folder. The script requires two arguments: hardware architecture and destination location. The value of <**architecture**> can be **x86**, **amd64**, or **arm** and <**destination**> is a path to a local directory. If the directory doesn't already exist, it will be created.

   Windows Command Prompt

```
copype.cmd <architecture> <destination>
```

For example, the following command copies **amd64** architecture files to the **C:\winpe_amd64** directory:

Windows Command Prompt

```
copype.cmd amd64 C:\winpe_amd64
```

The script creates the destination directory structure and copies all the necessary files for that architecture. In the previous example, the following directories are created:

Windows Command Prompt

```
C:\winpe_amd64
C:\winpe_amd64\fwfiles
C:\winpe_amd64\media
C:\winpe_amd64\mount
```

4. Mount the base Windows PE image (winpe.wim) to the \mount directory using the DISM tool. Mounting an image file unpacks the file contents into a folder so that you can make changes directly or by using tools such as DISM. See the following example.

Windows Command Prompt

```
dism.exe /mount-image /imagefile:c:\winpe_amd64\media\sources\boot.wim
/index:1 /mountdir:C:\winpe_amd64\mount
```

Verify that the message **The operation completed successfully** is displayed.

> **Note**
>
> To view currently mounted images, enter `dism.exe /get-MountedWiminfo`.

5. Map a network share to the root TFTP directory on the PXE/TFTP server and create a \Boot folder. Consult your TFTP server documentation to determine the root TFTP server directory, then enable sharing for this directory, and verify it can be accessed on the network. In the following example, the PXE server name is PXE-1 and the TFTP root directory is shared using a network path of **\\PXE-1\TFTPRoot**:

Windows Command Prompt

```
net.exe use y: \\PXE-1\TFTPRoot
y:
md Boot
```

6. Copy the PXE boot files from the mounted directory to the \boot folder. For example:

Windows Command Prompt

```
copy c:\winpe_amd64\mount\windows\boot\pxe\*.* y:\Boot
```

7. Copy the boot.sdi file to the PXE/TFTP server.

Windows Command Prompt

```
copy C:\winpe_amd64\media\boot\boot.sdi y:\Boot
```

8. Copy the bootable Windows PE image (boot.wim) to the \boot folder.

Windows Command Prompt

```
copy C:\winpe_amd64\media\sources\boot.wim y:\Boot
```

9. (Optional) Copy TrueType fonts to the \boot folder

Windows Command Prompt

```
copy C:\winpe_amd64\media\Boot\Fonts y:\Boot\Fonts
```

Step 2: Configure boot settings and copy the BCD file

1. Create a BCD store using bcdedit.exe:

Windows Command Prompt

```
bcdedit.exe /createstore c:\BCD
```

2. Configure RAMDISK settings:

Windows Command Prompt

```
bcdedit.exe /store c:\BCD /create {ramdiskoptions} /d "Ramdisk options"
bcdedit.exe /store c:\BCD /set {ramdiskoptions} ramdisksdidevice boot
bcdedit.exe /store c:\BCD /set {ramdiskoptions} ramdisksdipath
\Boot\boot.sdi
bcdedit.exe /store c:\BCD /create /d "winpe boot image" /application
osloader
```

The last command will return a GUID, for example:

Console

```
The entry {a4f89c62-2142-11e6-80b6-00155da04110} was successfully
created.
```

Copy this GUID for use in the next set of commands. In each command shown, replace "GUID1" with your GUID.

3. Create a new boot application entry for the Windows PE image:

Windows Command Prompt

```
bcdedit.exe /store c:\BCD /set {GUID1} device ramdisk=
[boot]\Boot\boot.wim,{ramdiskoptions}
bcdedit.exe /store c:\BCD /set {GUID1} path
\windows\system32\winload.exe
bcdedit.exe /store c:\BCD /set {GUID1} osdevice ramdisk=
[boot]\Boot\boot.wim,{ramdiskoptions}
bcdedit.exe /store c:\BCD /set {GUID1} systemroot \windows
bcdedit.exe /store c:\BCD /set {GUID1} detecthal Yes
bcdedit.exe /store c:\BCD /set {GUID1} winpe Yes
```

4. Configure BOOTMGR settings (remember to replace GUID1 in the third command with your GUID):

Windows Command Prompt

```
bcdedit.exe /store c:\BCD /create {bootmgr} /d "boot manager"
bcdedit.exe /store c:\BCD /set {bootmgr} timeout 30
bcdedit.exe /store c:\BCD -displayorder {GUID1} -addlast
```

5. Copy the BCD file to your TFTP server:

Windows Command Prompt

```
copy c:\BCD \\PXE-1\TFTPRoot\Boot\BCD
```

Your PXE/TFTP server is now configured. You can view the BCD settings that have been configured using the command bcdedit.exe /store <BCD file location> /enum all. See the following example. Note: Your GUID will be different than the one shown below.

Windows Command Prompt

```
C:\>bcdedit.exe /store C:\BCD /enum all
Windows Boot Manager
--------------------
identifier              {bootmgr}
description             boot manager
displayorder            {a4f89c62-2142-11e6-80b6-00155da04110}
timeout                 30

Windows Boot Loader
-------------------
identifier              {a4f89c62-2142-11e6-80b6-00155da04110}
device                  ramdisk=[boot]\Boot\boot.wim,{ramdiskoptions}
description             winpe boot image
osdevice                ramdisk=[boot]\Boot\boot.wim,{ramdiskoptions}
systemroot              \Windows
detecthal               Yes
winpe                   Yes

Setup Ramdisk Options
---------------------
identifier              {ramdiskoptions}
description             ramdisk options
ramdisksdidevice        boot
ramdisksdipath          \Boot\boot.sdi
```

Tip

If you start the PXE boot process, but receive the error **The boot configuration data for your PC is missing or contains error**, then verify that `\boot` directory is installed under the correct TFTP server root directory. In the example used here the name of this directory is TFTPRoot, but your TFTP server might be different.

PXE boot process summary

The following process summarizes the PXE client boot.

> **Note**
>
> The following assumes that the client and PXE server are on the same network/subnet/vlan or that PXE requests have been appropriately forwarded from the client to the PXE server using IP helpers configured in the router or switch. For more information about IP helpers, see Configuring Your Router to Forward Broadcasts.

1. A client contacts the PXE server. When the client is on a different network/subnet/vlan as the PXE server, the client is routed to the PXE server using the IP helpers.
2. The PXE server sends DHCP options 060 (client identifier **PXEClient**), 066 (boot server host name) and 067 (boot file name) to the client.
3. The client downloads `boot\PXEboot.n12` from the TFTP server based on DHCP option 067 boot file name value received from the PXE server.
4. `PXEboot.n12` immediately begins a network boot.
5. The client downloads `boot\bootmgr.exe` and the `boot\BCD` file from the TFTP server.

   > **Note**
   >
   > The BCD store must reside in the `\boot` directory on the TFTP server and must be named BCD.

6. `Bootmgr.exe` reads the BCD operating system entries and downloads `boot\boot.sdi` and the Windows PE image (`boot\boot.wim`). Optional files that can also be downloaded include TrueType fonts (`boot\Fonts\wgl4_boot.ttf`) and the hibernation state file (`\hiberfil.sys`) if these files are present.
7. `Bootmgr.exe` starts Windows PE by calling `winload.exe` within the Windows PE image.
8. Windows PE loads, a command prompt opens and `wpeinit.exe` is run to initialize Windows PE.
9. The Windows PE client provides access to tools like `imagex.exe`, `diskpart.exe`, and `bcdboot.exe` using the Windows PE command prompt. With the help of these tools

accompanied by a Windows 10 image file, the destination computer can be formatted properly to load a full Windows 10 operating system.

Related articles

Windows PE Walkthroughs

Feedback

Was this page helpful? Yes No

Provide product feedback

Windows Deployment Services (WDS) boot.wim support

Article • 07/19/2024 •

Applies to: ✅ Windows 11, ✅ Windows 10, ✅ Windows Server 2022, ✅ Windows Server 2019, ✅ Windows Server 2016

The operating system deployment functionality of Windows Deployment Services (WDS) is being partially deprecated. Starting with Windows 11, workflows that rely on **boot.wim** from installation media or on running Windows Setup in WDS mode is no longer supported.

When PXE booting from a WDS server that uses the **boot.wim** file from installation media as its boot image, Windows Setup automatically launches in WDS mode. This workflow is deprecated for Windows 11 and newer boot images. The following deprecation message is displayed:

> Windows Setup
>
> Windows Deployment Services client functionality is being partly deprecated. Please visit https://aka.ms/WDSSupport for more details on what is deprecated and what is still supported.

Deployment scenarios affected

The following table provides support details for specific deployment scenarios. Boot.wim is the `boot.wim` file obtained from the Windows source files for each specified version of Windows.

Expand table

Windows Version being deployed	Boot.wim from Windows 10	Boot.wim from Windows Server 2016	Boot.wim from Windows Server 2019	Boot.wim from Windows Server 2022	Boot.wim from Windows 11
Windows 11	Not supported, blocked.	Not supported, blocked.	Not supported, blocked.	Not supported, blocked.	Not supported, blocked.
Windows 10	Supported, using a boot image from	Supported, using a boot image from a	Supported, using a boot image from a	Not supported.	Not supported.

Windows Version being deployed	Boot.wim from Windows 10	Boot.wim from Windows Server 2016	Boot.wim from Windows Server 2019	Boot.wim from Windows Server 2022	Boot.wim from Windows 11
	matching or newer version.	currently supported version of Windows 10.	currently supported version of Windows 10.		
Windows Server 2025	Not supported.	Not supported.	Not supported.	Not supported.	Not supported.
Windows Server 2022	Deprecated, with a warning message.	Deprecated, with a warning message.	Deprecated, with a warning message.	Deprecated, with a warning message.	Not supported.
Windows Server 2019	Supported, using a boot image from a currently supported version of Windows 10.	Supported.	Supported.	Not supported.	Not supported.
Windows Server 2016	Supported, using a boot image from a currently supported version of Windows 10.	Supported.	Not supported.	Not supported.	Not supported.

Note

The following error message might be displayed when attempting to use **boot.wim** on WDS running on Windows Server 2025:

```
A media driver your computer needs is missing. This could be a DVD, USB or Hard disk driver. If you have a CD, DVD, or USB flash drive with the driver on it, please insert it now.
```

An error message is expected since using **boot.wim** on WDS running on Windows Server 2025 isn't supported.

Reason for the change

Alternatives to WDS, such as Microsoft Configuration Manager, provide a better, more flexible, and feature-rich experience for deploying Windows images.

Not affected

This change doesn't affect WDS PXE boot. WDS can still be used to PXE boot devices with custom boot images, but **boot.wim** can't be used as the boot image and run Windows Setup in WDS mode.

Windows Setup can still run from a network share. This change doesn't change Workflows that use a custom boot.wim, such as Microsoft Deployment Toolkit (MDT) or Microsoft Configuration Manager.

Summary

- Windows 11 workflows that rely on **boot.wim** from installation media are blocked. An end to end deployment of Windows 11 using only WDS can't be performed.
- This change doesn't affect Windows 10, Windows Server 2019, and previous operating system versions.
- Windows Server 2022 workflows that rely on **boot.wim** from installation media show a non-blocking deprecation notice. The notice can be dismissed, and currently the workflow isn't blocked.
- Windows Server workflows after Windows Server 2022 that rely on **boot.wim** from installation media are blocked.

If WDS is being used with **boot.wim** from installation media for end-to-end operating system deployment, and the OS version isn't supported, deprecated, or blocked, Microsoft recommends using deployment tools such as Microsoft Configuration Manager, or a non-Microsoft solution that uses a custom boot.wim image.

Related content

- Features removed or no longer developed starting with Windows Server 2022.
- Customize boot images with Configuration Manager.

Feedback

Was this page helpful? Yes No

Provide product feedback

Windows ADK for Windows scenarios for IT Pros

Article • 02/27/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

The Windows Assessment and Deployment Kit (Windows ADK) contains tools that IT Pros can use to deploy Windows. For an overview of what's new in the latest version of the Windows ADK, see What's new in the ADK tools. For the ADK reference content, see Desktop manufacturing.

Create a Windows image using command-line tools

DISM is used to mount and service Windows images.

Here are some things that can be done with DISM:

- Mount an offline image.
- Add and Remove Driver packages to an offline Windows Image.
- Enable or Disable Windows Features Using DISM.
- Add or Remove Packages Offline Using DISM.
- Add languages to Windows images.
- Preinstall Apps Using DISM.
- Change the Windows Image to a Higher Edition Using DISM.

Sysprep prepares a Windows installation for imaging and allows capturing a customized Windows installation.

Here are some things that can be done with Sysprep:

- Generalize a Windows installation.
- Customize the default user profile by using CopyProfile.
- Use answer files.

Windows PE (WinPE) is a small operating system used to boot a computer that doesn't have an operating system. Windows PE can be booted into to install a new operating system, recover data, or repair an existing operating system.

A WinPE image can be created using the article Create bootable Windows PE media. Types of bootable media include:

- Create a bootable Windows PE USB drive.

- Create a WinPE ISO, DVD, or CD.
- Create a Windows PE VHD to use with Hyper-V.

Windows Recovery Environment (Windows RE) is a recovery environment that can repair common operating system problems.

Here are some things that can be done with Windows RE:

- Customize Windows RE.
- Push-button reset.

Windows System Image Manager (WSIM) helps create answer files that change Windows settings and run scripts during Windows installation.

Here are some things that can be done with WSIM:

- Create or Open an Answer File.
- Add a Device Driver Path to an Answer File.
- Add a Package to an Answer File.
- Add a Custom Command to an Answer File.

For a list of settings that can be changed, see Unattended Windows Setup Reference.

Create a provisioning package using Windows ICD

Windows Imaging and Configuration Designer (ICD) streamlines the customizing and provisioning of a Windows for desktop editions (Home, Pro, Enterprise, and Education) or a Windows IoT Core (IoT Core) image. Creating, applying, and exporting provisioning packages with the Windows ICD is covered in the article Create a provisioning package.

IT Pro Windows deployment tools

There are also a few tools included in the Windows ADK that are specific to IT Pros:

- Volume Activation Management Tool (VAMT) Technical Reference
- User State Migration Tool (USMT) Technical Reference

Feedback

Was this page helpful? Yes No

Provide product feedback

User State Migration Tool (USMT) overview

Article • 03/27/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

The User State Migration Tool (USMT) can be used to streamline and simplify user state migration during large deployments of Windows operating systems. USMT captures user accounts, user files, operating system settings, and application settings, and then migrates them to a new Windows installation. USMT can be used for both PC replacement and PC refresh migrations. For more information, see Common migration scenarios.

USMT enables the following actions:

- Configure the migration according to the organization's business needs by using the migration rule (.xml) files to control exactly which files and settings are migrated and how they're migrated. For more information about how to modify these files, see USMT XML reference.
- Fit the customized migration into the automated deployment process by using the **ScanState** and **LoadState** tools, which control collecting and restoring the user files and settings. For more information, see User State Migration Tool (USMT) command-line syntax.
- Perform offline migrations. Migrations can be run offline by using the **ScanState** command in Windows Preinstallation Environment (WinPE) or migrations can be performed from previous installations of Windows contained in **Windows.old** directories. For more information about migration types, see Choose a migration store Type and Offline migration reference.

Benefits

USMT provides the following benefits to businesses that are deploying Windows operating systems:

- Safely migrates user accounts, operating system and application settings.
- Lowers the cost of deploying Windows by preserving user state.
- Reduces end-user downtime required to customize desktops and find missing files.
- Reduces help-desk calls.
- Reduces the time needed for the user to become familiar with the new operating system.

- Increases employee satisfaction with the migration experience.

Limitations

USMT is intended for administrators who are performing large-scale automated deployments. If the user states of only a few computers are being migrated, PCmover Express can be used. PCmover isn't a free utility. PCmover Express is a tool created by Microsoft's partner, Laplink.

There are some scenarios in which the use of USMT isn't recommended. These scenarios include:

- Migrations that require end-user interaction.
- Migrations that require customization on a machine-by-machine basis.

Download and install USMT

USMT is part of the Windows Assessment and Deployment Kit (Windows ADK) and can be downloaded from the following link:

Download and install the Windows ADK

Once the Windows ADK installer is downloaded, run the installer, and then in the **Select the features you want to install** page, make sure **User State Migration Tool (USMT)** is selected.

Related articles

- User State Migration Tool (USMT) technical reference.

Feedback

Was this page helpful? Yes No

Provide product feedback

Getting started with the User State Migration Tool (USMT)

Article • 03/27/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

This article outlines the general process to follow to migrate files and settings.

Step 1: Download and install USMT

USMT is part of the Windows Assessment and Deployment Kit (Windows ADK) and can be downloaded from the following link:

Download and install the Windows ADK

Once the Windows ADK installer is downloaded, run the installer, and then in the **Select the features you want to install** page, make sure **User State Migration Tool (USMT)** is selected.

Step 2: Plan the migration

1. Plan The Migration. Depending on whether the migration scenario is refreshing or replacing computers, an online migration or an offline migration can be chosen. Offline migrations can use either Windows Preinstallation Environment (WinPE) or the files in the **Windows.old** directory. For more information, see Common Migration Scenarios.

2. Determine What to Migrate. Data to consider migrating includes end-user information, applications settings, operating-system settings, files, folders, and registry keys.

3. Determine where to store data. Depending on the size of the migration store, data can be stored in one of the following locations:

   - Remotely.
   - Locally in a hard-link migration store or on a local external storage device.
   - Directly on the destination computer.

   For more information, see Choose a Migration Store Type.

4. Use the `/GenMigXML` command-line option to determine which files are included in the migration, and to determine whether any modifications are necessary. For more information, see ScanState Syntax

5. If necessary, modify copies of the `Migration.xml` and `MigDocs.xml` files and create custom **.xml** files. To modify the migration behavior, such as migrating the **Documents** folder but not the **Music** folder, custom **.xml** file can be created or modify the rules in the existing migration **.xml** files. The document finder, or `MigXmlHelper.GenerateDocPatterns` helper function, can be used to automatically find user documents on a computer without creating extensive custom migration **.xml** files.

   > **Important**
   >
   > Microsoft recommends to always make copies of the **.xml** files included in User State Migration Tool (USMT) and then modify the copies. Never modify the original **.xml** files.

   The `MigXML.xsd` file can be used to help write and validate the **.xml** files. For more information about how to modify these files, see USMT XML Reference.

6. Create a Config.xml File if to exclude any components from the migration. To create this file, run the `ScanState.exe` command with the following options:

   - /genconfig.
   - /i - as arguments specify the **.xml** files that are being used with `ScanState.exe`.

   For example, the following command creates a `Config.xml` file by using the `MigDocs.xml` and `MigApp.xml` files:

   Windows Command Prompt

   ```
   ScanState.exe /genconfig:Config.xml /i:MigDocs.xml /i:MigApp.xml /v:13
   /l:ScanState.log
   ```

7. Open the `Config.xml` that was generated in the previous step. Review the migration state of each of the components listed in the `Config.xml` file. If necessary, edit the `Config.xml` file and specify `migrate=no` for any components that don't need to be migrated.

Step 3: Collect files and settings from the source computer

1. Back up the source computer.

2. Close all applications. If some applications are running when the `ScanState.exe` command is run, USMT might not migrate all of the specified data. For example, if Microsoft Office Outlook is open, USMT might not migrate PST files.

> **Note**
>
> USMT fails if it can't migrate a file or setting unless the `/c` option is specified. When the `/c` option is specified, USMT ignores the errors, and logs an error every time that it encounters a file that is being used that USMT didn't migrate. The `<ErrorControl>` section in the `Config.xml` file can be used to specify which errors should be ignored, and which should cause the migration to fail.

3. To collect files and settings, run the `ScanState.exe` command on the source computer. All of the **.xml** files that the `ScanState.exe` command needs to use should be specified. For example,

Windows Command Prompt

```
ScanState.exe \\server\migration\mystore /config:Config.xml
/i:MigDocs.xml /i:MigApp.xml /v:13 /l:ScanState.log
```

> **Note**
>
> The `ScanState.exe` command must be run in **Administrator** mode on the source computer. To run in **Administrator** mode, right-click **Command Prompt**, and then select **Run As Administrator**. For more information about how the `ScanState.exe` command processes and stores the data, see How USMT Works.

4. To ensure that the created store isn't corrupted, run the `UsmtUtils.exe` command with the `/Verify` option.

Step 4: Prepare the destination computer and restore files and settings

1. Install the operating system on the destination computer.

2. Install all applications that were on the source computer. Although it isn't always required, Microsoft recommends installing all applications on the destination computer before restoring the user state. Installing all applications before restoring user state makes sure that migrated settings are preserved.

> **Note**
>
> The application version that is installed on the destination computer should be the same version as the one on the source computer. USMT doesn't support migrating the settings for an older version of an application to a newer version. The exception for this rule is Microsoft Office. USMT can migrate from an older version of Microsoft Office to a newer version of Microsoft Office.

3. Close all applications. If some applications are running when the `LoadState.exe` command runs, USMT might not migrate all of the specified data. For example, if Microsoft Office Outlook is open, USMT might not migrate PST files.

> **Note**
>
> Use `/c` to continue the migration if errors are encountered. Use the `<ErrorControl>` section in the `Config.xml` file to specify which errors should be ignored, and which errors should cause the migration to fail.

4. Run the `LoadState.exe` command on the destination computer. Specify the same set of **.xml** files that were specified when the `ScanState.exe` command was used. However, the `Config.xml` file doesn't always need to be specified. The `Config.xml` file only needs to be specified to exclude some of the files and settings that were migrated to the store. For example, the **Documents** folder was migrated to the store, but doesn't need to be migrated to the destination computer. For example, modify the `Config.xml` file and specify the updated file by using the `LoadState.exe` command. Then, the `LoadState.exe` command migrates only the files and settings that need to be migrated. For more information about how the `LoadState.exe` command processes and migrates data, see How USMT Works.

   For example, the following command migrates the files and settings:

   Windows Command Prompt

   ```
   LoadState.exe  \\server\migration\mystore /config:Config.xml
   ```

```
/i:MigDocs.xml /i:MigApp.xml /v:13 /l:LoadState.log
```

> **Note**
>
> Run the `LoadState.exe` command in administrator mode. To do this, right-click **Command Prompt**, and then select **Run As Administrator**.

5. Sign out after running the `LoadState.exe` command. Some settings, such as fonts, wallpaper, and screen saver settings, won't take effect until the next time that the user logs on.

Feedback

Was this page helpful? Yes No

Provide product feedback

Exclude files and settings

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

When the migration **.xml** files `MigApp.xml`, `MigDocs.xml`, and `MigUser.xml` are specified, the User State Migration Tool (USMT) migrates the settings and components listed, as discussed in What does USMT migrate? A custom **.xml** file can be created to further specify what to include or exclude in the migration. In addition, a `Config.xml` file can be created to exclude an entire component from a migration. However, users can't be excluded by using the migration **.xml** files or the `Config.xml` file. The only way to specify which users to include and exclude is by using the user options on the command line in the **ScanState** tool. For more information, see the User options section of the ScanState syntax article.

Methods to customize the migration and include and exclude files and settings include:

- Create a custom .xml file. The following elements can be used to specify what to exclude:
  - Include and exclude: The **<include>** and **<exclude>** elements can be used to exclude objects with conditions. For example, all files located in the `C:\` drive can be migrated, except any `.mp3` files. It's important to remember that Conflicts and precedence apply to these elements.
  - unconditionalExclude: The **<unconditionalExclude>** element can be used to globally exclude data. This element takes precedence over all other include and exclude rules in the **.xml** files. Therefore, this element excludes objects regardless of any other **<include>** rules that are in the **.xml** files. For example, all **.mp3** files can be excluded on the computer, or all files from C:\UserData can be excluded.
- Create a Config.xml file: A `Config.xml` file can be created and modified to exclude an entire component from the migration. For example, this file can be used to exclude the settings for one of the default applications. In addition, creating and modifying a `Config.xml` file is the only way to exclude the operating-system settings that are migrated to computers running Windows. Excluding components using this file is easier than modifying the migration **.xml** files because familiarity with the migration rules and syntax isn't required.

Create a custom .xml file

Microsoft recommends creating a custom **.xml** file instead of modifying the default migration **.xml** files. When a custom **.xml** file is used, the changes can be kept separate from the default **.xml** file, which makes it easier to track the modifications.

<include> and <exclude>

The migration **.xml** files, `MigApp.xml`, `MigDocs.xml`, and `MigUser.xml`, contain the **<component>** element, which typically represents a self-contained component or an application such as Microsoft Office Outlook and Word. To exclude the files and registry settings that are associated with these components, use the **<include>** and **<exclude>** elements. For example, these elements can be used to migrate all files and settings with pattern X except files and settings with pattern Y, where Y is more specific than X. For the syntax of these elements, see USMT XML Reference.

> **Note**
>
> If an **<exclude>** rule is specified, always specify a corresponding **<include>** rule. Otherwise, if an **<include>** rule isn't specified, the specific files or settings aren't included. They're already excluded from the migration. Thus, an unaccompanied **<exclude>** rule is unnecessary.

- Example 1: How to migrate all files from C:\ except .mp3 files
- Example 2: How to migrate all files located in C:\Data except files in C:\Data\tmp
- Example 3: How to exclude the files in a folder but include all subfolders
- Example 4: How to exclude a file from a specific folder
- Example 5: How to exclude a file from any location

Example 1: How to migrate all files from `C:\` except `.mp3` files

The following **.xml** file migrates all files located on the C: drive, except any **.mp3** files.

XML

```
<migration
urlid="http://www.microsoft.com/migration/1.0/migxmlext/mp3files">
    <!-- This component migrates all files except those with .mp3 extension-
->
    <component type="Documents" context="UserAndSystem">
```

```
        <displayName _locID="miguser.sharedvideo">MP3 Files</displayName>
        <role role="Data">
            <rules>
                <include filter='MigXmlHelper.IgnoreIrrelevantLinks()'>
                    <objectSet>
                        <pattern type="File">C:\* [*]</pattern>
                    </objectSet>
                </include>
                <exclude>
                    <objectSet>
                        <pattern type="File">C:\* [*.mp3]</pattern>
                    </objectSet>
                </exclude>
            </rules>
        </role>
    </component>
</migration>
```

Example 2: How to migrate all files located in `C:\Data` except files in `C:\Data\tmp`

The following **.xml** file migrates all files and subfolders in `C:\Data`, except the files and subfolders in `C:\Data\tmp`.

XML

```
<migration urlid="http://www.microsoft.com/migration/1.0/migxmlext/test">
    <component type="Documents" context="System">
        <displayName _locID="miguser.sharedvideo">Test
component</displayName>
        <role role="Data">
            <rules>
                <include>
                    <objectSet>
                        <pattern type="File">C:\Data\* [*]</pattern>
                    </objectSet>
                </include>
                <exclude>
                    <objectSet>
                        <pattern type="File"> C:\Data\temp\* [*]</pattern>
                    </objectSet>
                </exclude>
            </rules>
        </role>
    </component>
</migration>
```

Example 3: How to exclude the files in a folder but include all subfolders

The following **.xml** file migrates any subfolders in `C:\EngineeringDrafts`, but excludes all files that are in `C:\EngineeringDrafts`.

XML

```xml
<migration urlid="http://www.microsoft.com/migration/1.0/migxmlext/test">
    <component type="Documents" context="System">
        <displayName>Component to migrate all Engineering Drafts Documents
without subfolders</displayName>
        <role role="Data">
            <rules>
                <include>
                    <objectSet>
                        <pattern type="File"> C:\EngineeringDrafts\* [*]
</pattern>
                    </objectSet>
                </include>
                <exclude>
                    <objectSet>
                        <pattern type="File"> C:\EngineeringDrafts\ [*]
</pattern>
                    </objectSet>
                </exclude>
            </rules>
        </role>
    </component>
</migration>
```

Example 4: How to exclude a file from a specific folder

The following **.xml** file migrates all files and subfolders in `C:\EngineeringDrafts`, except for the `Sample.doc` file in `C:\EngineeringDrafts`.

XML

```xml
<migration urlid="http://www.microsoft.com/migration/1.0/migxmlext/test">
    <component type="Documents" context="System">
        <displayName>Component to migrate all Engineering Drafts Documents
except Sample.doc</displayName>
        <role role="Data">
            <rules>
                <include>
                    <objectSet>
                        <pattern type="File"> C:\EngineeringDrafts\* [*]
</pattern>
                    </objectSet>
```

```
                </include>
                <exclude>
                    <objectSet>
                        <pattern type="File"> C:\EngineeringDrafts\
[Sample.doc]</pattern>
                    </objectSet>
                </exclude>
            </rules>
        </role>
    </component>
</migration>
```

Example 5: How to exclude a file from any location

To exclude a Sample.doc file from any location on the C: drive, use the **<pattern>** element. If multiple files exist with the same name on the C: drive, all of these files are excluded.

```XML
<pattern type="File"> C:\* [Sample.doc] </pattern>
```

To exclude a Sample.doc file from any drive on the computer, use the **<script>** element. If multiple files exist with the same name, all of these files are excluded.

```XML
<script>MigXmlHelper.GenerateDrivePatterns("* [sample.doc]", "Fixed")
</script>
```

Examples of how to use XML to exclude files, folders, and registry keys

Here are some examples of how to use XML to exclude files, folders, and registry keys. For more info, see USMT XML Reference

Example 1: How to exclude all `.mp3` files

The following **.xml** file excludes all `.mp3` files from the migration:

```XML
<migration
urlid="http://www.microsoft.com/migration/1.0/migxmlext/excludefiles">
```

```
    <component context="System" type="Documents">
        <displayName>Test</displayName>
        <role role="Data">
            <rules>
                <unconditionalExclude>
                    <objectSet>
                        <script>MigXmlHelper.GenerateDrivePatterns ("*
[*.mp3]", "Fixed")</script>
                    </objectSet>
                </unconditionalExclude>
            </rules>
        </role>
    </component>
</migration>
```

Example 2: How to exclude all of the files on a specific drive

The following **.xml** file excludes only the files located on the C: drive.

XML

```
<migration
urlid="http://www.microsoft.com/migration/1.0/migxmlext/allfiles">
    <component type="Documents" context="System">
        <displayName>Test</displayName>
        <role role="Data">
            <rules>
                <unconditionalExclude>
                    <objectSet>
                        <pattern type="File">c:\*[*]</pattern>
                    </objectSet>
                </unconditionalExclude>
            </rules>
        </role>
    </component>
</migration>
```

Example 3: How to exclude registry keys

The following **.xml** file unconditionally excludes the `HKEY_CURRENT_USER` registry key and all of its subkeys.

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<migration urlid="http://www.microsoft.com/migration/1.0/migxmlext/miguser">
    <component type="Documents" context="User">
        <displayName>Test</displayName>
        <role role="Data">
```

```
            <rules>
                <include>
                    <objectSet>
                        <pattern type="Registry">HKCU\testReg[*]</pattern>
                    </objectSet>
                </include>
                <unconditionalExclude>
                    <objectSet>
                        <pattern type="Registry">HKCU\*[*]</pattern>
                    </objectSet>
                </unconditionalExclude>
            </rules>
        </role>
    </component>
</migration>
```

Example 4: How to Exclude `C:\Windows` and `C:\Program Files`

The following **.xml** file unconditionally excludes the system folders of `C:\Windows` and `C:\Program Files`. All `*.docx`, `*.xls` and `*.ppt` files aren't migrated because the **<unconditionalExclude>** element takes precedence over the **<include>** element.

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<migration urlid="http://www.microsoft.com/migration/1.0/migxmlext/miguser">
    <component type="Documents" context="System">
        <displayName>Test</displayName>
        <role role="Data">
            <rules>
                <include>
                    <objectSet>
                        <script>MigXmlHelper.GenerateDrivePatterns ("*
[*.doc]", "Fixed")</script>
                        <script>MigXmlHelper.GenerateDrivePatterns ("*
[*.xls]", "Fixed")</script>
                        <script>MigXmlHelper.GenerateDrivePatterns ("*
[*.ppt]", "Fixed")</script>
                    </objectSet>
                </include>
                <unconditionalExclude>
                    <objectSet>
                        <pattern type="File">C:\Program Files\* [*]
</pattern>
                        <pattern type="File">C:\Windows\* [*]</pattern>
                    </objectSet>
                </unconditionalExclude>
            </rules>
        </role>
```

```
    </component>
</migration>
```

Create a Config XML File

A `Config.xml` file can be created and modified to exclude components from the migration. Excluding components using this file is easier than modifying the migration **.xml** files because familiarity with the migration rules and syntax isn't required. `Config.xml` is an optional file that can be created using the `/genconfig` command-line option with the **ScanState** tool. For example, the `Config.xml` file can be used to exclude the settings for one of the default applications. In addition, creating and modifying this file is the only way to exclude the operating-system settings that are migrated to computers running Windows.

- **To exclude the settings for a default application:** Specify `migrate="no"` for the application under the <**Applications**> section of the `Config.xml` file.
- **To exclude an operating system setting:** Specify `migrate="no"` for the setting under the <**WindowsComponents**> section.
- **To exclude the Documents folder:** Specify `migrate="no"` for the **Documents** folder under the <**Documents**> section. Any <**include**> rules in the **.xml** files are still applied. For example, if a rule exists that includes all the **.docx** files in the **Documents** folder, then **.docx** files are still migrated. However, any additional files that aren't **.docx** aren't migrated.

For more information, see Config.xml File.

> ⓘ **Note**
>
> To exclude a component from the `Config.xml` file, set the **migrate** value to **"no"**. Deleting the XML tag for the component from the `Config.xml` file doesn't exclude the component from the migration.

Related articles

- Customize USMT XML files.
- USMT XML reference.

Feedback

Was this page helpful? Yes No

Provide product feedback

Extract files from a compressed USMT migration store

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

When files and settings are migrated during a typical PC-refresh migration, a compressed migration store file is usually created on the intermediate store. This migration store is a single image file that contains all files being migrated as well as a catalog file. To protect the compressed file, it can be encrypted by using different encryption algorithms. When the file is migrated back to the source computer after the operating system is installed, the **UsmtUtils** command can be run with the `/extract` option to recover the files from the compressed migration store. The **UsmtUtils** command with the `/extract` option can also be used any time data needs to be recovered from a migration store.

Options used with the `/extract` option can specify:

- The cryptographic algorithm that was used to create the migration store.
- The encryption key or the text file that contains the encryption key.
- Include and exclude patterns for selective data extraction.

In addition, the file patterns that need to be extracted can be specified by using the `/i` option to include file patterns or the `/e` option to exclude file patterns. When both the `/i` option and the `/e` option are used in the same command, include patterns take precedence over exclude patterns. The `/i` and the `/e` options are different from the include and exclude rules used in the **ScanState** and **LoadState** tools.

To run the UsmtUtils tool with the /extract option

To extract files from the compressed migration store onto the destination computer, use the following UsmtUtils syntax:

Windows Command Prompt

```
UsmtUtils.exe /extract <filePath> <destinationPath> [/i:<includePattern>]
[/e:<excludePattern>] [/l:<logfile>] [/decrypt[:<AlgID>] {/key:<keystring> |
/keyfile:<filename>}] [/o]
```

Where the placeholders have the following values:

- **<USMTpath>** is the location where the USMT files and tools are saved.
- **<filePath>** is the location of the migration store.
- **<destination path>** is the location of the file where the **/extract** option should put the extracted migration store contents.
- **<includePattern>** specifies the pattern for the files to include in the extraction.
- **<excludePattern>** specifies the pattern for the files to omit from the extraction.
- **<AlgID>** is the cryptographic algorithm that was used to create the migration store on the `ScanState.exe` command line.
- **<logfile>** is the location and name of the log file.
- **<keystring>** is the encryption key that was used to encrypt the migration store.
- **<filename>** is the location and name of the text file that contains the encryption key.

To extract all files from a compressed migration store

To extract everything from a compressed migration store to a file on the `C:\` drive, enter:

Windows Command Prompt

```
UsmtUtils.exe /extract D:\MyMigrationStore\USMT\store.mig C:\ExtractedStore
```

To extract specific file types from an encrypted compressed migration store

To extract specific files, such as `.txt` and `.pdf` files, from an encrypted compressed migration store, enter:

Windows Command Prompt

```
UsmtUtils.exe /extract D:\MyMigrationStore\USMT\store.mig /i:"*.txt,*.pdf"
C:\ExtractedStore /decrypt /keyfile:D:\encryptionKey.txt
```

In this example, the file is encrypted and the encryption key is located in a text file called encryptionKey.

To extract all but one, or more, file types from an encrypted compressed migration store

To extract all files except for one file type, such as `.exe` files, from an encrypted compressed migration store, enter:

Windows Command Prompt

```
UsmtUtils.exe /extract D:\MyMigrationStore\USMT\store.mig /e:*.exe
C:\ExtractedStore /decrypt:AES_128 /key:password /l:C:\usmtutilslog.txt
```

To extract file types using the include pattern and the exclude pattern

When files are extracted from a compressed migration store, both the include and the exclude patterns can be used at the same time. Files of one type can be excluded while files of another type can be included. For example:

Windows Command Prompt

```
UsmtUtils.exe /extract D:\MyMigrationStore\USMT\store.mig /i:myProject.*
/e:*.exe C:\ExtractedStore /o
```

In this example, if there's a **myProject.exe** file, the file is also extracted because the include pattern option takes precedence over the exclude pattern option.

Related articles

- UsmtUtils syntax.
- Return codes.
- Verify the condition of a compressed migration store.

Feedback

Was this page helpful? Yes No

Provide product feedback

Include Files and Settings

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

When the migration **.xml** files are specified, User State Migration Tool (USMT) migrates the settings and components specified in What does USMT migrate?. To include additional files and settings, Microsoft recommends creating a custom **.xml** file, and then include this file when using both the `ScanState.exe` and `LoadState.exe` commands. Creating a custom **.xml** file allows changes to be kept separate from the default **.xml** files. Creating a custom **.xml** file makes it easier to track modifications.

Migrate a single registry key

The following **.xml** file migrates a single registry key.

XML

```xml
<migration urlid="http://www.microsoft.com/migration/1.0/migxmlext/test">
     <component type="Application" context="System">
          <displayName>Component to migrate only registry value
string</displayName>
          <role role="Settings">
          <rules>
               <include>
                    <objectSet>
                         <pattern
type="Registry">HKLM\Software\Microsoft\Windows\CurrentVersion\Internet
Settings\Cache [Persistent]</pattern>
                    </objectSet>
               </include>
          </rules>
          </role>
     </component>
</migration>
```

Migrate a specific folder

The following examples show how to migrate a folder from a specific drive, and from any location on the computer.

Migrate a folder from a specific drive

- **Including subfolders.** The following **.xml** file migrates all files and subfolders from `C:\EngineeringDrafts` to the destination computer.

```XML
<migration
urlid="http://www.microsoft.com/migration/1.0/migxmlext/test">
<component type="Documents" context="System">
  <displayName>Component to migrate all Engineering Drafts Documents
including subfolders</displayName>
  <role role="Data">
    <rules>
      <include>
        <objectSet>
          <pattern type="File">C:\EngineeringDrafts\* [*]</pattern>
        </objectSet>
      </include>
    </rules>
  </role>
</component>
</migration>
```

- **Excluding subfolders.** The following **.xml** file migrates all files from `C:\EngineeringDrafts`, but it doesn't migrate any subfolders within `C:\EngineeringDrafts`.

```XML
<migration
urlid="http://www.microsoft.com/migration/1.0/migxmlext/test">
<component type="Documents" context="System">
  <displayName>Component to migrate all Engineering Drafts Documents
without subfolders</displayName>
  <role role="Data">
    <rules>
      <include>
        <objectSet>
          <pattern type="File"> C:\EngineeringDrafts\ [*]</pattern>
        </objectSet>
      </include>
    </rules>
  </role>
</component>
</migration>
```

Migrate a folder from any location

The following **.xml** file migrates all files and subfolders of the `EngineeringDrafts` folder from any drive on the computer. If multiple folders exist with the same name, then all files with this name are migrated.

```XML
<migration urlid="http://www.microsoft.com/migration/1.0/migxmlext/test">
<component type="Documents" context="System">
  <displayName>Component to migrate all Engineering Drafts Documents folder
on any drive on the computer </displayName>
  <role role="Data">
    <rules>
         <include>
            <objectSet>
         <script>MigXmlHelper.GenerateDrivePatterns ("\EngineeringDrafts\*
[*] ", "Fixed")</script>
         <script>MigXmlHelper.GenerateDrivePatterns ("*\EngineeringDrafts\*
[*] ", "Fixed")</script>
       </objectSet>
          </include>
    </rules>
  </role>
</component>
</migration>
```

The following **.xml** file migrates all files and subfolders of the `EngineeringDrafts` folder from any location on the `C:\` drive. If multiple folders exist with the same name, they're all migrated.

```XML
<migration urlid="http://www.microsoft.com/migration/1.0/migxmlext/test">
<component type="Documents" context="System">
  <displayName>Component to migrate all Engineering Drafts Documents
EngineeringDrafts folder from where ever it exists on the C: drive
</displayName>
  <role role="Data">
    <rules>
         <include>
            <objectSet>
<pattern type="File"> C:\*\EngineeringDrafts\* [*]</pattern>
<pattern type="File"> C:\EngineeringDrafts\* [*]</pattern>
       </objectSet>
          </include>
    </rules>
  </role>
</component>
</migration>
```

Migrate a file type into a specific folder

The following **.xml** file migrates `.mp3` files located in the specified drives on the source computer into the `C:\Music` folder on the destination computer.

```XML
<migration urlid="http://www.microsoft.com/migration/1.0/migxmlext/test">
<component type="Documents" context="System">
  <displayName>All .mp3 files to the Documents folder</displayName>
  <role role="Data">
    <rules>
      <include>
        <objectSet>
          <script>MigXmlHelper.GenerateDrivePatterns ("* [*.mp3]", "Fixed")
</script>
        </objectSet>
      </include>
      <!-- Migrates all the .mp3 files in the store to the C:\Music folder
during LoadState -->
      <locationModify script="MigXmlHelper.Move('C:\Music')">
        <objectSet>
          <script>MigXmlHelper.GenerateDrivePatterns ("* [*.mp3]", "Fixed")
</script>
        </objectSet>
      </locationModify>
    </rules>
  </role>
</component>
</migration>
```

Migrate a specific file

The following examples show how to migrate a file from a specific folder, and how to migrate a file from any location.

- **To migrate a file from a folder.** The following **.xml** file migrates only the `Sample.doc` file from `C:\EngineeringDrafts` on the source computer to the destination computer.

  ```XML
  <migration
  urlid="http://www.microsoft.com/migration/1.0/migxmlext/test">
  <component type="Documents" context="System">
    <displayName>Component to migrate all Engineering Drafts
  Documents</displayName>
    <role role="Data">
      <rules>
        <include>
          <objectSet>
            <pattern type="File"> C:\EngineeringDrafts\ [Sample.doc]
  </pattern>
          </objectSet>
        </include>
  ```

```
    </rules>
  </role>
</component>
</migration>
```

- **To migrate a file from any location.** To migrate the `Sample.doc` file from any location on the `C:\` drive, use the **<pattern>** element, as the following example shows. If multiple files exist with the same name on the `C:\` drive, all of files with this name are migrated.

  XML

  ```
  <pattern type="File"> C:\* [Sample.doc] </pattern>
  ```

  To migrate the Sample.doc file from any drive on the computer, use <script> as the following example shows. If multiple files exist with the same name, all files with this name are migrated.

  XML

  ```
  <script>MigXmlHelper.GenerateDrivePatterns("* [sample.doc]", "Fixed")
  </script>
  ```

Related articles

- Customize USMT XML files.
- Custom XML examples.
- Conflicts and precedence.
- USMT XML reference.

Feedback

Was this page helpful? Yes No

Provide product feedback

Migrate Application Settings

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

A custom **.xml** file can be created to migrate specific line-of-business application settings or to change the default migration behavior of the User State Migration Tool (USMT). For **ScanState** and **LoadState** to use this file, the custom **.xml** file must be specified on both command lines.

This article defines how to author a custom migration **.xml** file that migrates the settings of an application that isn't migrated by default using `MigApp.xml`. The settings should be migrated after the application is installed, but before the user runs the application for the first time.

This article doesn't contain information about how to migrate applications that store settings in an application-specific store, only the applications that store the information in files or in the registry. It also doesn't contain information about how to migrate the data that users create using the application. For example, if the application creates **.doc** files using a specific template, this article doesn't discuss how to migrate the **.doc** files and templates themselves.

Before beginning

A test computer that contains the operating system of the source computers should be identified. The test computer should also have the applications whose settings need to be migrated. For example, if migrating from Windows 10 to Windows 11, install Windows 10 on the test computer and then install the applications.

Step 1: Verify that the application is installed on the source computer, and that it's the same version as the version to be installed on the destination computer

Before USMT migrates the settings, check whether the application is installed on the source computer, and that it's the correct version. If the application isn't installed on the source computer, USMT still spends time searching for the application's settings. More importantly, if USMT collects settings for an application that isn't installed, it could migrate settings that cause the destination computer to function incorrectly. Also determine whether there's more than one version of the application because the new

version could store the settings in a different location. Mismatched application versions could lead to unexpected results on the destination computer.

There are many ways to detect if an application is installed. The best practice is to check for an application uninstall key in the registry. The computer can then be searched for the executable file that installed the application. It's important to check for both of these items, because sometimes different versions of the same application share the same uninstall key. Even if the key is there, it could correspond to a different version of the application that is wanted.

Check the registry for an application uninstall key

When many applications are installed, especially those installed using the Microsoft Windows Installer technology, an application uninstall key is created under:

`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall`

For example, when Adobe Acrobat Reader 7 is installed, it creates a key named:

`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall \{AC76BA86-7AD7-1033-7B44-A70000000000}`

Therefore, if a computer contains this key, then Adobe Acrobat Reader 7 is installed on the computer. The existence of a registry key can be checked using the `DoesObjectExist` helper function.

The **Uninstall** registry key for a particular application can be found under the following registry key:

`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall`

To find the **Uninstall** key for a specific application, search for one of the following items under the **Uninstall** registry key:

- Name of the application.
- Name of the application executable file.
- Name of the company that makes the application.

To search the registry, use the Registry Editor `Regedit.exe`. `Regedit.exe` is located in the path stored in `%SystemRoot%`, normally `C:\Windows`.

Check the file system for the application executable file

The application binaries for the executable that installed the application should also be checked. To check for application binaries, determine where the application is installed and what the name of the executable is. Most applications store the installation location of the application binaries in the registry. The registry should be searched on one of the following items until the registry value that contains the installation path is found:

- The name of the application.
- The name of the application executable.
- The name of the company that makes the application.

Once the path to the application executable is determined, the `DoesFileVersionMatch` helper function can be used to check for the correct version of the application executable. For an example of how to use the `DoesFileVersionMatch` helper function, see the Windows Live™ Messenger section of the `MigApp.xml` file.

Step 2: Identify settings to collect and determine where each setting is stored on the computer

Next, go through the user interface and make a list of all of the available settings. The list can be reduced if there are settings that don't need to be migrated. To determine where each setting is stored, change the setting. As the setting is changed, monitor the activity on the registry and the file system through a tool such as Process Monitor. The binary files and registry settings that are created when the application is installed don't need to be migrated. When the application is reinstalled onto the destination computer, it recreates those settings. Only the customized settings need to be migrated.

How to determine where each setting is stored

1. Download a file and registry monitoring tool, such as Process Monitor (Procmon), from the Sysinternals Web site.

2. Shut down as many applications as possible to limit the registry and file system activity on the computer.

3. Filter the output of the tools so it only displays changes being made by the application.

> ⓘ **Note**

> Most applications store their settings under the user profile. That is, the settings stored in the file system are under the `%UserProfile%` directory, and the settings stored in the registry are under the `HKEY_CURRENT_USER` hive. For these applications, the output of the file and registry monitoring tools can be filtered to show activity only under these locations. This filtering considerably reduces the amount of output that needs to be examined.

4. Start the monitoring tools, change a setting, and look for registry and file system writes that occurred when the setting was changed. Make sure the changes made actually take effect. For example, if changing a setting in Microsoft Word by selecting a check box in the **Options** dialog box, the change typically doesn't take effect until the dialog box is closed by selecting **OK**.

5. When the setting is changed, note the changes to the file system and registry. There could be more than one file or registry values for each setting. The minimal set of file and registry changes that are required to change this setting should be identified. This set of files and registry keys is what needs to be migrated in order to migrate the setting.

> **Note**
>
> Changing an application setting invariably leads to writing to registry keys. If possible, filter the output of the file and registry monitor tool to display only writes to files and registry keys/values.

Step 3: Identify how to apply the gathered settings

If the version of the application on the source computer is the same as the one on the destination computer, then the collected files and registry keys don't need to be modified. By default, USMT migrates the files and registry keys from the source location to the corresponding location on the destination computer. For example, if a file was collected from the `C:\Users\User1\Documents` folder and the profile directory on the destination computer is located at `D:\Users\User1`, then USMT automatically migrates the file to `D:\Users\User1\Documents`. However, the location of some settings might need to be modified in the following three cases:

Case 1: The version of the application on the destination computer is newer than the one on the source computer

In this case, the newer version of the application might be able to read the settings from the source computer without modification. That is, the data collected from an older version of the application is sometimes compatible with the newer version of the application. However, the setting location might need to be modified if either of the following conditions is true:

- **The newer version of the application has the ability to import settings from an older version.** This mapping usually happens the first time a user runs the newer version after the settings are migrated. Some applications import settings automatically after settings are migrated. However, other applications only import settings if the application was upgraded from the older version. When the application is upgraded, a set of files and/or registry keys is installed that indicates the older version of the application was previously installed. If a clean installation of the newer version is performed, the computer doesn't contain these files and registry keys. If the files and registry keys aren't present, the mapping doesn't occur. In order to trick the newer version of the application into initiating this import process, the migration script might need to create these files and/or registry keys on the destination computer.

  To identify which files and/or registry keys/values need to be created so that the import works:

  1. Upgrade the older version of the application to the newer one.
  2. Monitor the changes made to the file system and registry by using the same process described in How to determine where each setting is stored.

  Once the set of files that the computer needs is known, the **<addObjects>** element can be used to add them to the destination computer.

- **The newer version of the application can't read settings from the source computer and it's also unable to import the settings into the new format.** In this case, create a mapping for each setting from the old locations to the new locations. To create the mapping, determine where the newer version stores each setting using the process described in How to determine where each setting is stored. After creating the mapping, apply the settings to the new location on the destination computer using the **<locationModify>** element, and the `RelativeMove` and `ExactMove` helper functions.

Case 2: The destination computer already contains settings for the application

Microsoft recommends migrating the settings after the application is installed, but before the user runs the application for the first time. Microsoft recommends this process because this process ensures that there are no settings on the destination computer when the settings are migrated. If the application must be installed before the migration, any existing settings should be deleted using the **<destinationCleanup>** element. If for any reason the settings need to be preserved that are on the destination computer, the **<merge>** element and `DestinationPriority` helper function can be used.

Case 3: The application overwrites settings when installed

Microsoft recommends migrating the settings after the application is installed, but before the user runs the application for the first time. Microsoft recommends this process because this process ensures that there are no settings on the destination computer when the settings are migrated. Also, when some applications are installed, they overwrite any existing settings that are on the computer. In this scenario, if the data was migrated before the application was installed, the customized settings would be overwritten. This scenario is common for applications that store settings in locations that are outside of the user profile (typically these settings are settings that apply to all users). These universal settings are sometimes overwritten when an application is installed, and they're replaced by default values. To avoid this problem, these applications must be installed before migrating the files and settings to the destination computer. By default with USMT, data from the source computer overwrites data that already exists in the same location on the destination computer.

Step 4: Create the migration XML component for the application

After completing steps 1 through 3, create a custom migration **.xml** file that migrates the application based on the updated information. The `MigApp.xml` file can be used as a model because it contains examples of many of the concepts discussed in this article. Also see Custom XML Examples for another sample **.xml** file.

> **Note**
>
> Microsoft recommends creating a separate **.xml** file instead of adding a script to the `MigApp.xml` file. A separate **.xml** file is recommended because the `MigApp.xml`

file is a large file and it's difficult to read and edit. In addition, if USMT is reinstalled, the `MigApp.xml` file is overwritten with the default version of the file and the customized version is lost.

> **ⓘ Important**
>
> Some applications store information in the user profile, such as application installation paths, the computer name, etc. Application information stored in the user profile shouldn't be migrated and should be excluded from the migration.

The script should do the following actions:

1. Check if the correct version of the application is installed:
   - Search for the installation uninstall key under `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall` using the `DoesObjectExist` helper function.
   - Check for the correct version of the application executable file using the `DoesFileVersionMatch` helper function.
2. If the correct version of the application is installed, then ensure that each setting is migrated to the appropriate location on the destination computer.
   - If the versions of the applications are the same on both the source and destination computers, migrate each setting using the **<include>** and **<exclude>** elements.
   - If the version of the application on the destination computer is newer than the one on the source computer, and the application can't import the settings, the script should either:
     a. Add the set of files that trigger the import using the **<addObjects>** element.
     b. Create a mapping that applies the old settings to the correct location on the destination computer using the **<locationModify>** element, and the `RelativeMove` and `ExactMove` helper functions.
   - If the application must be installed before migrating the settings, delete any settings that are already on the destination computer using the **<destinationCleanup>** element.

For information about the **.xml** elements and helper functions, see XML Elements Library.

Step 5: Test the application settings migration

On a test computer, install the operating system that will be installed on the destination computers. For example, if planning on migrating from Windows 10 to Windows 11, install Windows 11, and then install the application in Windows 11. Next, run **LoadState** on the test computer and verify that all settings migrate. Make corrections if necessary and repeat the process until all the necessary settings are migrated correctly.

To speed up the time it takes to collect and migrate the data, only one user can be migrated at a time. All other components can be excluded from the migration except the application that is being tested. To specify only **User1** in the migration, enter:

Windows Command Prompt

```
/ue:*\* /ui:user1
```

For more information, see the Exclude files and settings article and the User options section in the ScanState syntax article. To troubleshoot a problem, check the progress log, the **ScanState** log, and the **LoadState** log. The logs contain warnings and errors that could point to problems with the migration.

Related articles

- USMT XML reference.
- Conflicts and precedence.
- XML elements library.
- Log files.

Feedback

Was this page helpful? Yes No

Provide product feedback

Migrate EFS files and certificates

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

This article describes how to migrate Encrypting File System (EFS) certificates. For more information about the `/efs` option, see Encrypted file options in ScanState syntax.

To migrate EFS files and certificates

Encrypting File System (EFS) certificates are migrated automatically. However, by default, the User State Migration Tool (USMT) fails if an encrypted file is found unless the `/efs` option is specified. Therefore when a device has EFS encrypted files, the `/efs` option must be specified with any one of the following parameters:

- `abort`
- `skip`
- `decryptcopy`
- `copyraw`
- `hardlink`

when running the `ScanState.exe` command to migrate the encrypted files. Then, when the `LoadState.exe` command is run on the destination computer, the encrypted file and the EFS certificate are automatically migrated.

> ⓘ **Note**
>
> The `/efs` options aren't used with the `LoadState.exe` command.

Before using the **ScanState** tool for a migration that includes encrypted files and EFS certificates, all files in an encrypted folder must also be encrypted. Otherwise, remove the encryption attribute from folders that contain unencrypted files. If the encryption attribute is removed from a file but not from the parent folder, the file is encrypted during the migration using the credentials of the account used to run the **LoadState** tool.

The Cipher.exe tool can be run at a Windows command prompt to review and change encryption settings on files and folders. For example, to remove encryption from a folder, at a command prompt enter:

```
Windows Command Prompt
```

```
cipher.exe /D /S:<PATH>
```

where <*Path*> is the full path of the topmost parent directory where the encryption attribute is set.

Related articles

- What does USMT migrate?.
- Identify file types, files, and folders.

Feedback

Was this page helpful? Yes No

Provide product feedback

Migrate User Accounts

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

By default, all users are migrated. The only way to specify which users to include and exclude is on the command line by using the ScanState User options and the LoadState User options. Users can't be specified in the migration XML files or by using the `Config.xml` file.

To migrate all user accounts and user settings

Links to detailed explanations of commands are available in the Related articles section.

1. Sign into the source computer as an administrator.
2. Enter the following `ScanState.exe` command line in a command prompt window:

   Windows Command Prompt

   ```
   ScanState.exe \\server\share\migration\mystore /i:MigDocs.xml
   /i:MigApp.xml /o
   ```

3. Sign into the destination computer as an administrator.
4. Enter one of the following `LoadState.exe` command lines in a command prompt window:

   - If migrating domain accounts, enter:

     Windows Command Prompt

     ```
     LoadState.exe  \\server\share\migration\mystore /i:MigDocs.xml
     /i:MigApp.xml
     ```

   - If migrating local accounts along with domain accounts, enter:

     Windows Command Prompt

     ```
     LoadState.exe  \\server\share\migration\mystore /i:MigDocs.xml
     /i:MigApp.xml /lac /lae
     ```

     Note

> The `/lae` option doesn't need to be specified, which enables the account that was created with the `/lac` option. Instead, create a disabled local account by specifying only the `/lac` option, and then a local administrator needs to enable the account on the destination computer.

To migrate two domain accounts (User1 and User2)

Links to detailed explanations of commands are available in the Related articles section.

1. Sign into the source computer as an administrator.
2. Enter the following `ScanState.exe` command line in a command prompt window:

   Windows Command Prompt

   ```
   ScanState.exe \\server\share\migration\mystore /ue:*\*
   /ui:contoso\user1 /ui:fabrikam\user2 /i:MigDocs.xml /i:MigApp.xml /o
   ```

3. Sign into the destination computer as an administrator.
4. Enter the following `LoadState.exe` command line in a command prompt window:

   Windows Command Prompt

   ```
   LoadState.exe  \\server\share\migration\mystore /i:MigDocs.xml
   /i:MigApp.xml
   ```

To migrate two domain accounts (User1 and User2) and move both accounts from the Contoso domain to the Fabrikam domain

Links to detailed explanations of commands are available in the Related articles section.

1. Sign into the source computer as an administrator.
2. Enter the following `ScanState.exe` command line in a command prompt window:

   Windows Command Prompt

```
ScanState.exe \\server\share\migration\mystore /ue:*\*
/ui:contoso\user1 /ui:contoso\user2 /i:MigDocs.xml /i:MigApp.xml /o
```

3. Sign into the destination computer as an administrator.

4. Enter the following `LoadState.exe` command line in a command prompt window:

Windows Command Prompt

```
LoadState.exe  \\server\share\migration\mystore
/mu:contoso\user1:fabrikam\user1 /mu:contoso\user2:fabrikam\user2
/i:MigDocs.xml /i:MigApp.xml
```

Related articles

- Identify users.
- ScanState syntax.
- LoadState syntax.

Feedback

Was this page helpful? Yes No

Provide product feedback

Reroute Files and Settings

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

To reroute files and settings, create a custom **.xml** file and specify the **.xml** file name on both the `ScanState.exe` and `LoadState.exe` command-lines. The custom **.xml** file enables keeping changes separate from the default **.xml** files, so that it's easier to track modifications.

Reroute a folder

The following custom **.xml** file migrates the directories and files from `C:\EngineeringDrafts` into the **Documents** folder of every user. **%CSIDL_PERSONAL%** is the virtual folder representing the **<User>** desktop item, which is equivalent to **CSIDL_MYDOCUMENTS**.

XML

```xml
<migration urlid="http://www.microsoft.com/migration/1.0/migxmlext/test">
<component type="Documents" context="User">
  <displayName>Engineering Drafts Documents to Personal Folder</displayName>
  <role role="Data">
    <rules>
      <!-- Migrate all directories and files present in c:\EngineeringDrafts
folder -->
      <include>
        <objectSet>
          <pattern type="File">C:\EngineeringDrafts\* [*]</pattern>
        </objectSet>
      </include>
      <!-- This migrates all files and directories from C:\EngineeringDrafts
to every user's personal folder.-->
      <locationModify
script="MigXmlHelper.RelativeMove('C:\EngineeringDrafts','%CSIDL_PERSONAL%')
">
        <objectSet>
          <pattern type="File">C:\EngineeringDrafts\* [*]</pattern>
        </objectSet>
      </locationModify>
    </rules>
  </role>
</component>
</migration>
```

Reroute a specific file type

The following custom **.xml** file reroutes **.mp3** files located in the fixed drives on the source computer into the `C:\Music` folder on the destination computer.

XML

```
<migration urlid="http://www.microsoft.com/migration/1.0/migxmlext/test">
<component type="Documents" context="System">
  <displayName>All .mp3 files to the Documents folder</displayName>
  <role role="Data">
    <rules>
      <include>
        <objectSet>
          <script>MigXmlHelper.GenerateDrivePatterns ("* [*.mp3]", "Fixed")
</script>
        </objectSet>
      </include>
      <!-- Migrates all the .mp3 files in the store to the C:\Music folder
during LoadState -->
      <locationModify script="MigXmlHelper.Move('C:\Music')">
        <objectSet>
          <script>MigXmlHelper.GenerateDrivePatterns ("* [*.mp3]", "Fixed")
</script>
        </objectSet>
      </locationModify>
    </rules>
  </role>
</component>
</migration>
```

Reroute a specific file

The following custom **.xml** file migrates the `Sample.doc` file from `C:\EngineeringDrafts` into the **Documents** folder of every user. **%CSIDL_PERSONAL%** is the virtual folder representing the **<User>** desktop item, which is equivalent to **CSIDL_MYDOCUMENTS**.

XML

```
<migration urlid="http://www.microsoft.com/migration/1.0/migxmlext/test">
<component type="Documents" context="User">
<displayName>Sample.doc into the Documents folder</displayName>
    <role role="Data">
      <rules>
        <include>
          <objectSet>
                 <pattern type="File"> C:\EngineeringDrafts\ [Sample.doc]
</pattern>
          </objectSet>
        </include>
       <locationModify
script="MigXmlHelper.RelativeMove('C:\EngineeringDrafts','%CSIDL_PERSONAL%')
```

```
">
        <objectSet>
                <pattern type="File"> C:\EngineeringDrafts\ [Sample.doc]
</pattern>
        </objectSet>
       </locationModify>
      </rules>
    </role>
</component>
</migration>
```

Related articles

- Customize USMT XML files.
- Conflicts and precedence.
- USMT XML reference.

Feedback

Was this page helpful? Yes No

Provide product feedback

Verify the condition of a compressed migration store

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

When files and settings are migrated during a typical PC-refresh migration, the user state is normally stored in a compressed folder on the intermediate store. This compressed folder, also called the compressed migration store, is a single image file that contains:

- All of the files being migrated.
- The user's settings.
- A catalog file that contains metadata for all files in the migration store.

When the `LoadState.exe` command is run to load the data from these files to the destination computer, **LoadState** requires a valid catalog file in order to open the migration store. The `UsmtUtils.exe` command can be run with the `/verify` option to determine whether the compressed migration store is intact, or whether it contains corrupted files or a corrupted catalog. The `/verify` option should be run on the migration store before overwriting the original user-state files and settings.

When the `/verify` option is used, what type of information to report in the **UsmtUtils** log file can be specified. These report types are:

- **Catalog**: Displays the status of only the catalog file.
- **All**: Displays the status of all files, including the catalog file.
- **Failure only**: Displays only the files that are corrupted.

The following sections demonstrate how to run the `UsmtUtils.exe` command with the `/verify` option, and how to specify the information to display in the **UsmtUtils** log file.

The UsmtUtils syntax for the /verify option

To verify the condition of a compressed migration store, use the following UsmtUtils syntax:

> UsmtUtils.exe /verify[:<*reportType*>] <*filePath*> [/l:<*logfile*>] [/decrypt [:<*AlgID*>] {/key:<*keystring*> | /keyfile:<*filename*>}]

Where the placeholders have the following values:

- <*USMTpath*> is the location where the USMT files and tools are saved.
- <*reportType*> specifies whether to report on all files, corrupted files only, or the status of the catalog.
- <*filePath*> is the location of the compressed migration store.
- <*logfile*> is the location and name of the log file.
- <*AlgID*> is the cryptographic algorithm that was used to create the migration store on the `ScanState.exe` command line.
- <*keystring*> is the encryption key that was used to encrypt the migration store.
- <*filename*> is the location and name of the text file that contains the encryption key.

To verify that the migration store is intact

To verify whether the migration store is intact or whether it contains corrupted files or a corrupted catalog, enter:

Windows Command Prompt

```
UsmtUtils.exe /verify D:\MyMigrationStore\store.mig
```

Because no report type is specified, **UsmtUtils** displays the default summary report.

To verify the status of only the catalog file

To verify whether the catalog file is corrupted or intact, enter:

Windows Command Prompt

```
UsmtUtils.exe /verify:catalog D:\MyMigrationStore\store.mig
```

To verify the status of all files

To verify whether there are any corrupted files in the compressed migration store, and to specify the name and location of the log file, enter:

Windows Command Prompt

```
UsmtUtils.exe /verify:all D:\MyMigrationStore\store.mig /decrypt
/l:D:\UsmtUtilsLog.txt`
```

In addition to verifying the status of all files, this example decrypts the files. Because no encryption algorithm is specified, **UsmtUtils** uses the default 3DES cryptographic algorithm.

To verify the status of the files and return only the corrupted files

In this example, the log file only lists the files that became corrupted during the **ScanState** process. This list includes the catalog file if the catalog file is also corrupted.

Windows Command Prompt

```
UsmtUtils.exe /verify:failureonly D:\MyMigrationStore\USMT\store.mig
/decrypt:AES_192 /keyfile:D:\encryptionKey.txt
```

This example also decrypts the files by specifying the cryptographic algorithm and the location of the file that contains the encryption key.

Next steps

If the `/verify` option indicates that there are corrupted files in the migration store, the `/extract` option in the **UsmtUtils** tool can be used to recover data from some corrupted stores. For more information, see Extract files from a compressed USMT migration store.

Related articles

- UsmtUtils syntax.
- Return codes.

Feedback

Was this page helpful? Yes No

Provide product feedback

USMT requirements

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

Supported operating systems

The User State Migration Tool (USMT) doesn't have any explicit RAM or CPU speed requirements for either the source or destination computers. If the computer complies with the system requirements of the operating system, it also complies with the requirements for USMT. An intermediate store location large enough to hold all of the migrated data and settings is needed. The same amount of hard disk space is also needed on the destination computer for the migrated files and settings.

The following table lists the operating systems supported in USMT.

Expand table

Operating Systems	ScanState (Source Device)	LoadState (Destination Device)
Windows 7	✅	❌
Windows 8	✅	❌
Windows 10	✅	✅
Windows 11	✅	✅

> **Note**
>
> - 32-bit operating system can be migrated to a 64-bit operating system. However, a 64-bit operating system can't be migrated to a 32-bit operating system.
> - Gathering data from a source device using **ScanState** for a version of Windows that is out of support is supported. However, restoring data to a destination device using **LoadState** to a version of Windows that is out of support isn't supported.

Unsupported scenarios

- USMT doesn't support any of the Windows Server operating systems.
- USMT doesn't support Microsoft Entra joined devices as either a source or destination device.
- USMT might work with Microsoft Entra hybrid joined devices, but it's not a tested scenario so therefore unsupported.
- USMT doesn't support migrating settings for Microsoft Store apps.
- USMT shouldn't be used for migrating between previous versions of Windows. USMT is only meant to:
  - Migrate to a currently supported version of Windows
  - Migrate between currently supported versions of Windows, assuming the version of Windows being migrated to is newer or the same as the previous version of Windows being migrated from.

For more information about previous releases of the USMT tools, see User State Migration Tool (USMT) overview.

Windows PE

- **Must use latest version of Windows PE.** For more info, see What's New in Windows PE.

Credentials

- **Run as administrator.**

  When the **ScanState** and **LoadState** tools are run, they must be run from an elevated command prompt to ensure that all specified users are migrated. If USMT isn't run from an elevated prompt, only the user profile that is logged on is included in the migration.

To open an elevated command prompt:

1. Select **Start**.
2. Enter `cmd` in the search function.
3. **cmd** or **Command Prompt** is displayed.
4. Right-click **cmd** or **Command Prompt**, and then select **Run as administrator**.
5. If the current user isn't already an administrator, it prompts to enter administrator credentials.

> ⓘ **Important**

USMT must run using an account with full administrative permissions, including the following privileges:

- SeBackupPrivilege (Back up files and directories)
- SeDebugPrivilege (Debug programs)
- SeRestorePrivilege (Restore files and directories)
- SeSecurityPrivilege (Manage auditing and security log)
- SeTakeOwnership Privilege (Take ownership of files or other objects)

Config.xml

Specify the `/c` option and <ErrorControl> settings in the `Config.xml` file

USMT fails if it can't migrate a file or setting, unless the `/c` option is specified. When the `/c` option is specified, USMT logs an error each time it encounters a file that is in use that didn't migrate, but the migration isn't to be interrupted. In USMT, which types of errors should allow the migration to continue and which should cause the migration to fail can be specified in the `Config.xml` file. For more information about error reporting, and the <**ErrorControl**> element, see Config.xml file, Log files, and XML elements library.

LoadState

Install applications before running the LoadState command

Install all applications on the destination computer before restoring the user state. Installing applications before running the `LoadState.exe` command ensures that migrated settings are preserved.

Hard-disk requirements

Ensure that there's enough available space in the migration-store location and on the source and destination computers. For more information, see Estimate Migration Store Size.

User prerequisites

This documentation assumes that IT professionals using USMT understand command-line tools. The documentation also assumes that IT professionals using USMT to author MigXML rules understand the following concepts:

- The navigation and hierarchy of the Windows registry.
- The files and file types that applications use.
- The methods to extract application and setting information manually from applications created by internal software-development groups and non-Microsoft software publishers.
- XML-authoring basics.

Related articles

- Plan the migration.
- Estimate migration store size.
- User State Migration Tool (USMT) overview articles.

Feedback

Was this page helpful? Yes No

Provide product feedback

USMT best practices

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

This article discusses general and security-related best practices when using User State Migration Tool (USMT).

General best practices

- **Install applications before running the LoadState tool.**

  Though it isn't always essential, it's best practice to install all applications on the destination computer before restoring the user state. Installing applications before restoring user state helps ensure that migrated settings are preserved.

- **Don't use MigUser.xml and MigDocs.xml together.**

  If both `MigUser.xml` and `MigDocs.xml` are used together, some migrated files can be duplicated if conflicting instructions are given about target locations. The `/genmigxml` command-line option can be used to determine which files are included in the migration, and to determine if any modifications are necessary. For more information, see Identify file types, files, and folders.

- **Use MigDocs.xml for a better migration experience.**

  If the data set is unknown or if many files are stored outside of the standard user-profile folders, the `MigDocs.xml` file is a better choice than the `MigUser.xml` file, because the `MigDocs.xml` file gathers a broader scope of data. The `MigDocs.xml` file migrates folders of data based on location, and on registered file type by querying the registry for registered application extensions. The `MigUser.xml` file migrates only the files with the specified file extensions.

- **Close all applications before running either the ScanState or LoadState tools.**

  Although using the `/vsc` switch can allow the migration of many files that are open with another application, it's a best practice to close all applications in order to ensure all files and settings migrate. Without the `/vsc` or `/c` switch, USMT fails when it can't migrate a file or setting. When the `/c` option is used, USMT ignores any files or settings that it can't migrate and log an error each time.

- **Log off after running the LoadState.**

Some settings, such as fonts, wallpaper, and screensaver settings, won't take effect until the next time the user logs on. For this reason, sign out after running the **LoadState** tool.

- **Managed environment.**

  To create a managed environment, all of the end user's documents can be moved into the **Documents** folder (%CSIDL_PERSONAL%). Microsoft recommends migrating files into the smallest-possible number of folders on the destination computer. Minimizing folders helps to clean up files on the destination computer if the `LoadState.exe` command fails before completion.

- **Chkdsk.exe.**

  Microsoft recommends running **Chkdsk.exe** before running the **ScanState** and **LoadState** tools. **Chkdsk.exe** creates a status report for a hard disk drive and lists and corrects common errors. For more information about the **Chkdsk.exe** tool, see Chkdsk.

- **Migrate in groups.**

  If the migration is performed while users are using the network, it's best to migrate user accounts in groups. To minimize the effect on network performance, determine the size of the groups based on the size of each user account. Migrating in phases also allows making sure each phase is successful before starting the next phase. When this method is, any necessary modifications can be made to the plan between groups.

Security best practices

As the authorized administrator, it's the responsibility to protect the privacy of the users and maintain security during and after the migration. In particular, the following issues must be considered:

- **Encrypting File System (EFS).**

  Take extreme caution when migrating encrypted files, because the end user doesn't need to be logged on to capture the user state. By default, USMT fails if an encrypted file is found. For specific instructions about EFS best practices, see Migrate EFS Files and Certificates.

  ⓘ **Note**

> If an encrypted file is migrated without also migrating the certificate, end users won't be able to access the file after the migration.

- **Encrypt the store.**

  Consider using the `/encrypt` option with the `ScanState.exe` command and the `/decrypt` option with the `LoadState.exe` command. However, use extreme caution with this set of options, because anyone who has access to the `ScanState.exe` command-line script also has access to the encryption key.

- **Virus Scan.**

  Microsoft recommends to scan both the source and destination computers for viruses before running USMT. In addition, the destination computer image should be scanned. To help protect data from viruses, Microsoft strongly recommends running an antivirus utility before migration.

- **Maintain security of the file server and the deployment server.**

  Microsoft recommends managing the security of the file and deployment servers. It's important to make sure that the file server where the store is saved is secure. The deployment server must also be secured to ensure that the user data that is in the log files isn't exposed. Microsoft also recommends to only transmit data over a secure network connection, such as a virtual private network. For more information about network security, see Microsoft Security Compliance Manager.

- **Password Migration.**

  To ensure the privacy of the end users, USMT doesn't migrate passwords, including passwords for applications or mapped network drives. It's important to make sure that end users know their passwords.

- **Local Account Creation.**

  Before local accounts are migrated, see the Migrating Local Accounts section in the Identify Users article.

XML file best practices

- **Specify the same set of mig*.xml files in both the ScanState and the LoadState tools.**

If a particular set of mig*.xml files are used with the **ScanState** tool, either called through the `/auto` option, or individually through the `/i` option, then the same option should be used to call the exact same mig*.xml files in the **LoadState** tool.

- **The <CustomFileName> in the migration urlid should match the name of the file.**

  Although it isn't a requirement, it's good practice for **<CustomFileName>** to match the name of the file. For example, the following example is from the `MigApp.xml` file:

  XML

  ```xml
  <?xml version="1.0" encoding="UTF-8"?>
  <migration
  urlid="http://www.microsoft.com/migration/1.0/migxmlext/migapp">
  ```

- **Use the XML Schema (MigXML.xsd) when authoring .xml files to validate syntax.**

  The `MigXML.xsd` schema file shouldn't be included on the command line or in any of the **.xml** files.

- **Use the default migration XML files as models.**

  To create a custom **.xml** file, migration **.xml** files can be used as models to create customized versions. If user data files need to be migrated, model the custom **.xml** file on `MigUser.xml`. To migrate application settings, model the custom **.xml** file on the `MigApp.xml` file.

- **Consider the impact on performance when using the <context> parameter.**

  The migration performance can be affected when the **<context>** element is used with the **<component>** element. For example, when encapsulating logical units of file- or path-based **<include>** and **<exclude>** rules.

  In the **User** context, a rule is processed one time for each user on the system.

  In the **System** context, a rule is processed one time for the system.

  In the **UserAndSystem** context, a rule is processed one time for each user on the system and one time for the system.

  ⓘ Note

> The number of times a rule is processed doesn't affect the number of times a file is migrated. The USMT migration engine ensures that each file migrates only once.

- **Microsoft recommends to create a separate .xml file instead of adding .xml code to one of the existing migration .xml files.**

  For example, for code that migrates the settings for an application, the code shouldn't just be added to the `MigApp.xml` file.

- **Custom .xml files shouldn't be created to alter the operating system settings that are migrated.**

  Manifest files determine what settings are migrated. Manifest files can't be modified. Since manifest files can't be modified, to exclude certain operating system settings from the migration, create and modify a `Config.xml` file instead.

- **The asterisk (*) wildcard character can be used in any migration XML file that is created.**

  > **Note**
  >
  > The question mark isn't valid as a wildcard character in USMT **.xml** files.

Related articles

- Migration store encryption.
- Plan the migration.

Feedback

Was this page helpful? Yes No

Provide product feedback

How USMT works

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

USMT includes two tools that migrate settings and data: **ScanState** and **LoadState**. **ScanState** collects information from the source computer, and **LoadState** applies that information to the destination computer.

> ⓘ **Note**
>
> For more information about how USMT processes the rules and the XML files, see Conflicts and precedence.

The ScanState process

When the **ScanState** tool runs on the source computer, it goes through the following process:

1. It parses and validates the command-line parameters, creates the `ScanState.log` file, and then begins logging.

2. It collects information about all of the migration components that need to be migrated. A *migration component* is a logical group of files, registry keys, and values. For example, the set of files, registry keys, and values that store the settings of Adobe Acrobat is grouped into a single migration component.

   There are three types of components:

   - Components that migrate the operating system settings.
   - Components that migrate application settings.
   - Components that migrate users' files.

   The **ScanState** tool collects information about the application settings and user data components from the **.xml** files that are specified on the command line.

   In currently supported versions of Windows, the manifest files control how the operating-system settings are migrated. These files can't be modified. To exclude certain operating-system settings, a `Config.xml` file must be created and modified.

3. **ScanState** determines which user profiles should be migrated. By default, all user profiles on the source computer are migrated. However, users can be included and

excluded using the User options. The System profile and the Public profile in a source computer running currently supported versions of Windows is always migrated, and these profiles can't be excluded from the migration.

4. In the **Scanning** phase, **ScanState** does the following for each user profile selected for migration:

   a. For each component, **ScanState** checks the type of the component. If the current user profile is the system profile and the component type is **System** or **UserAndSystem**, the component is selected for this user. Otherwise, the component is ignored. Alternatively, if the current user profile isn't the system profile and the component type is **User** or **UserAndSystem**, the component is selected for this user. Otherwise, this component is ignored.

   > ⓘ **Note**
   >
   > From this point on, **ScanState** doesn't distinguish between components that migrate operating-system settings, components that migrate application settings, and components that migrate users' files. **ScanState** processes all components in the same way.

   b. Each component that is selected in the previous step is processed further. Any profile-specific variables (such as **CSIDL_PERSONAL**) are evaluated in the context of the current profile. For example, if the profile that is being processed belongs to **User1**, then **CSIDL_PERSONAL** would expand to `C:\Users\User1\Documents`, assuming that the user profiles are stored in the `C:\Users` directory.

   c. For each selected component, **ScanState** evaluates the **<detects>** section. If the condition in the **<detects>** section evaluates to false, the component isn't processed any further. Otherwise, the processing of this component continues.

   d. For each selected component, **ScanState** evaluates the **<rules>** sections. For each **<rules>** section, if the current user profile is the system profile and the context of the **<rules>** section is **System** or **UserAndSystem**, the rule is processed further. Otherwise, this rule is ignored. Alternatively, if the current user profile isn't the system profile and the context of the **<rules>** section is **User** or **UserAndSystem**, the rule is processed further. Otherwise, this rule is ignored.

   e. **ScanState** creates a list of migration units that need to be migrated by processing the various subsections under this **<rules>** section. Each unit is

collected if the unit is mentioned in an **<include>** subsection, as long as there isn't a more specific rule for it in an **<exclude>** subsection in the same **<rules>** section. For more information about precedence in the **.xml** files, see Conflicts and precedence.

In addition, any migration unit (such as a file, registry key, or set of registry values) that is in an <UnconditionalExclude> section isn't migrated.

> **Note**
>
> **ScanState** ignores some subsections such as <destinationCleanup> and <locationModify>. These sections are evaluated only on the destination computer.

5. In the **Collecting** phase, **ScanState** creates a central list of the migration units by combining the lists that were created for each selected user profile.

6. In the **Saving** phase, **ScanState** writes the migration units that were collected to the store location.

> **Note**
>
> **ScanState** doesn't modify the source computer in any way.

The LoadState process

The **LoadState** process is similar to the **ScanState** process. The **ScanState** tool collects migration units such as file, registry key, or registry values from the source computer and saves them to the store. Similarly, the **LoadState** tool collects migration units from the store and applies them to the destination computer.

1. **ScanState** parses and validates the command-line parameters, creates the `ScanState.log` file, and then begins logging.

2. **LoadState** collects information about the migration components that need to be migrated.

   LoadState obtains information for the application-settings components and user-data components from the migration **.xml** files that are specified by the `LoadState.exe` command.

In currently supported versions of Windows, the manifest files control how the operating-system settings are migrated. These files can't be modified. To exclude certain operating-system settings, a `Config.xml` file must be created and modified.

3. **LoadState** determines which user profiles should be migrated. By default, all user profiles present on the source computer are migrated. However, users can be included and excluded using the User options. The System profile and the Public profile in a source computer running currently supported versions of Windows is always migrated and these profiles can't be excluded from the migration.

   - If local user accounts are being migrated and if the accounts don't already exist on the destination computer, the `/lac` command-line option must be used. If the `/lac` option isn't specified, any local user accounts that aren't already present on the destination computer, aren't migrated.

   - When specified with the `LoadState.exe` command, the `/md` and `/mu` options are processed to rename the user profile on the destination computer.

   - For each user profile selected from the store, **LoadState** creates a corresponding user profile on the destination computer. The destination computer doesn't need to be connected to the domain for domain user profiles to be created. If USMT can't determine a domain, it attempts to apply the settings to a local account. For more information, see Identify Users.

4. In the **Scanning** phase, **LoadState** does the following for each user profile:

   a. For each component, **LoadState** checks the type of the component. If the current user profile is the system profile and the component type is **System** or **UserAndSystem**, the component is selected for this user. Otherwise, the component is ignored. Alternatively, if the current user profile isn't the system profile and the component type is **User** or **UserAndSystem**, the component is selected for this user. Otherwise, this component is ignored.

   > ⓘ **Note**
   >
   > From this point on, **LoadState** doesn't distinguish between components that migrate operating-system settings, components that migrate application settings, and components that migrate users' files. **LoadState** evaluates all components in the same way.

   b. Each component that is selected is processed further. Any profile-specific variables (such as **CSIDL_PERSONAL**) are evaluated in the context of the current

profile. For example, if the profile being processed belongs to **User1**, then **CSIDL_PERSONAL** would expand to `C:\Users\User1\Documents` (assuming that the user profiles are stored in the `C:\Users` directory).

> **Note**
>
> **LoadState** ignores the **<detects>** section specified in a component. At this point, all specified components are considered to be detected and are selected for migration.

c. For each selected component, **LoadState** evaluates the **<rules>** sections. For each **<rules>** section, if the current user profile is the system profile and the context of the **<rules>** section is **System** or **UserAndSystem**, the rule is processed further. Otherwise, this rule is ignored. Alternatively, if the current user profile isn't the system profile and the context of the **<rules>** section is **User** or **UserAndSystem**, the rule is processed further. Otherwise, this rule is ignored.

d. **LoadState** creates a central list of migration units by processing the various subsections under the **<rules>** section. Each migration unit that is in an **<include>** subsection is migrated as long, as there isn't a more specific rule for it in an **<exclude>** subsection in the same **<rules>** section. For more information about precedence, see Conflicts and precedence.

e. **LoadState** evaluates the destination computer-specific subsections, for example, the **<destinationCleanup>** and **<locationModify>** subsections.

f. If the destination computer is running a currently supported version of Windows, then the migunits that were collected by **ScanState** using downlevel manifest files are processed by **LoadState** using the corresponding Component Manifest from the downlevel Windows version. The downlevel manifest files aren't used during **LoadState**.

> **Important**
>
> For **LoadState** to use the **.xml** files, it's important to specify them with the `LoadState.exe` command. Otherwise, any destination-specific rules, such as **<locationModify>**, in these **.xml** files are ignored, even if the same **.xml** files were provided when the `ScanState.exe` command ran.

5. In the **Apply** phase, **LoadState** writes the migration units that were collected to the various locations on the destination computer. If there are conflicts and there isn't a **<merge>** rule for the object, the default behavior for the registry is for the source to overwrite the destination. The default behavior for files is for the source to be renamed incrementally, for example, OriginalFileName(1).OriginalExtension. Some settings, such as fonts, wallpaper, and screen-saver settings, don't take effect until the next time the user logs on. For this reason, sign out when the `LoadState.exe` command actions are finished.

Related articles

- User State Migration Tool (USMT) command-line syntax.

Feedback

Was this page helpful? Yes No

Provide product feedback

Plan the migration

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

Before using the User State Migration Tool (USMT) to perform a migration, Microsoft recommends that to plan the migration carefully. Planning can help the migration proceed smoothly and can reduce the risk of migration failure.

In migration planning, both organizations and individuals must first identify what to migrate, including user settings, applications and application settings, and personal data files and folders. Identifying the applications to migrate is especially important to avoid capturing data about applications that might be phased out.

One of the most important requirements for migrating settings and data is restoring only the information that the destination computer requires. Although the data that is captured on the source computer might be more comprehensive than the restoration data for backup purposes, restoring data or settings for applications that aren't installed on the destination system is redundant. Restoring data or settings for applications that aren't installed can also introduce instability in a newly deployed computer.

In this section

Expand table

Link	Description
Common migration scenarios	Determine whether to perform a refresh migration or a replace migration.
What does USMT migrate?	Learn which applications, user data, and operating system components USMT migrates.
Choose a migration store type	Choose an uncompressed, compressed, or hard-link migration store.
Determine what to migrate	Identify user accounts, application settings, operating system settings, and files that need to be migrated inside the organization.
Test the migration	Test the migration before deploying Windows to all users.

Related articles

- USMT XML reference.

Feedback

Was this page helpful? Yes No

Provide product feedback

Common Migration Scenarios

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

The User State Migration Tool (USMT) can be used when hardware and/or operating system upgrades are planned for a large number of computers. USMT manages the migration of an end-user's digital identity by capturing from a source computer the following user's items:

- Operating-system settings.
- Application settings.
- Personal files.

Once these items are capture, they're reinstalled on a destination computer after the upgrade completes.

One common scenario is when the operating system is upgraded on existing hardware without the hardware being replaced. This scenario is referred to as **PC-refresh**. A second common scenario is known as **PC replacement**, where one piece of hardware is being replaced, typically by newer hardware and a newer operating system.

PC-refresh

The following diagram shows a PC-refresh migration, also known as a computer refresh migration. First, the administrator migrates the user state from a source computer to an intermediate store. After the administrator installs the operating system, they migrate the user state back to the source computer.

Scenario One: PC-refresh offline using Windows PE and a hard-link migration store

An organization receives funds to update the operating system on all of its computers in the accounting department to the latest supported version of Windows. Each employee keeps the same computer, but the operating system on each computer will be updated. In this scenario, the update is being handled offline, without a network connection. An administrator uses Windows Preinstallation Environment (WinPE) and a hard-link migration store to save each user state to their respective computer.

1. On each computer, the administrator boots the machine into WinPE and runs the **ScanState** command-line tool, specifying the `/hardlink /nocompress` command-line options. **ScanState** saves the user state to a hard-link migration store on each computer, improving performance by minimizing network traffic and minimizing migration failures on computers with limited space available on the hard drive.

2. On each computer, the administrator installs the organization's standard operating environment (SOE) which includes the latest supported version of Windows and other organization applications.

3. The administrator runs the **LoadState** command-line tool on each computer. **LoadState** restores each user state back to each computer.

Scenario Two: PC-refresh using a compressed migration store

An organization receives funds to update the operating system on all of its computers to the latest supported version of Windows. Each employee keeps the same computer, but the operating system on each computer will be updated. In this scenario, an administrator uses a compressed migration store to save the user states to a server.

1. The administrator runs the **ScanState** command-line tool on each computer. **ScanState** saves each user state to a server.

2. On each computer, the administrator installs the organization's standard SOE that includes the latest supported version of Windows and other organization applications.

3. The administrator runs the **LoadState** command-line tool on each source computer, and **LoadState** restores each user state back to the computer.

Scenario Three: PC-refresh using a hard-link migration store

An organization receives funds to update the operating system on all of its computers to the latest supported version of Windows. Each employee keeps the same computer, but the operating system on each computer will be updated. In this scenario, an administrator uses a hard-link migration store to save each user state to their respective computer.

1. The administrator runs the **ScanState** command-line tool on each computer, specifying the `/hardlink /nocompress` command-line options. **ScanState** saves the user state to a hard-link migration store on each computer, improving performance by minimizing network traffic and minimizing migration failures on computers with limited space available on the hard drive.

2. On each computer, the administrator installs the organization's SOE that includes the latest supported version of Windows and other organization applications.

3. The administrator runs the **LoadState** command-line tool on each computer. **LoadState** restores each user state back on each computer.

Scenario Four: PC-refresh using Windows.old folder and a hard-link migration store

An organization decides to update the operating system on all of its computers to the latest supported version of Windows. Each employee keeps the same computer, but the operating system on each computer will be updated. In this scenario, an administrator uses **Windows.old** and a hard-link migration store to save each user state to their respective computer.

1. The administrator clean installs the latest supported version of Windows on each computer. During the install, they make sure that the **Windows.old** directory is created by taking the following actions:

    - Performing the install without formatting or repartitioning the disk.
    - Selecting a partition that contains the previous version of Windows.

2. On each computer, the administrator installs the organization's SOE that includes organization applications.

3. The administrator runs the **ScanState** and **LoadState** command-line tools successively on each computer while specifying the `/hardlink /nocompress` command-line options.

PC-replacement

The following diagram shows a PC-replacement migration. First, the administrator migrates the user state from the source computer to an intermediate store. After the administrator installs the operating system on the destination computer, they migrate the user state from the store to the destination computer.

Scenario One: Offline migration using Windows PE and an external migration store

An organization is allocating 20 new computers to users in the accounting department. The users each have a source computer with their files and settings. In this scenario, migration is being handled offline, without a network connection.

1. On each source computer, an administrator boots the machine into WinPE and runs **ScanState** to collect the user state to either a server or an external hard disk.

2. On each new computer, the administrator installs the organization's SOE that includes the latest supported version of Windows and other organization applications.

3. On each of the new computers, the administrator runs the **LoadState** tool, restoring each user state from the migration store to one of the new computers.

Scenario Two: Manual network migration

An organization receives 50 new laptops for their managers and needs to reallocate 50 older laptops to new employees. In this scenario, an administrator runs the **ScanState** tool from the cmd prompt on each computer to collect the user states and save them to a server in a compressed migration store.

1. The administrator runs the **ScanState** tool on each of the manager's old laptops, and saves each user state to a server.

2. On the new laptops, the administrator installs the organization's SOE, which includes the latest supported version of Windows and other organization applications.

3. The administrator runs the **LoadState** tool on the new laptops to migrate the managers' user states to the appropriate computer. The new laptops are now ready for the managers to use.

4. On the old computers, the administrator installs the organization's SOE, which includes the latest supported version of Windows, Microsoft Office, and other organization applications. The old computers are now ready for the new employees to use.

Scenario Three: Managed network migration

An organization is allocating 20 new computers to users in the accounting department. The users each have a source computer that contains their files and settings. An administrator uses a management technology such as a sign-in script or a batch file to run **ScanState** on each source computer to collect the user states and save them to a server in a compressed migration store.

1. On each source computer, the administrator runs the **ScanState** tool using Microsoft Configuration Manager, Microsoft Deployment Toolkit (MDT), a sign-in script, a batch file, or a non-Microsoft management technology. **ScanState** collects the user state from each source computer and then saves it to a server.

2. On each new computer, the administrator installs the organization's SOE, which includes the latest supported version of Windows and other organization applications.

3. On each of the new computers, the administrator runs the **LoadState** tool using Microsoft Configuration Manager, a sign-in script, a batch file, or a non-Microsoft

management technology. **LoadState** migrates each user state from the migration store to one of the new computers.

Related articles

- Plan the migration.
- Choose a migration store type.
- Offline migration reference.

Feedback

Was this page helpful? Yes No

Provide product feedback

What does USMT migrate?

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

Default migration scripts

The User State Migration Tool (USMT) is designed so that an IT engineer can precisely define migrations using the USMT **.xml** scripting language. USMT provides the following sample scripts:

- **MigApp.XML** - Rules to migrate application settings.
- **MigDocs.XML** - Rules that use the **MigXmlHelper.GenerateDocPatterns** helper function, which can be used to automatically find user documents on a computer without the need to author extensive custom migration **.xml** files.
- **MigUser.XML** - Rules to migrate user profiles and user data.

  `MigUser.xml` gathers everything in a user's profile and then does a file extension-based search of most of the system for other user data. If data doesn't match either of these criteria, the data isn't migrated. Usually, this file describes a core migration.

  The following data doesn't migrate with `MigUser.xml`:
  - Files outside the user profile that don't match one of the file extensions in `MigUser.xml`.
  - Access control lists (ACLs) for folders outside the user profile.

User data

This section describes the user data that USMT migrates by default, using the `MigUser.xml` file. It also defines how to migrate access control lists (ACLs).

- **Folders from each user profile.** When the `MigUser.xml` file is specified, USMT migrates everything in a user's profiles including the following folder items:
  - Documents.
  - Videos.
  - Music.
  - Pictures.

- Desktop files.
- Start menu.
- Quick Launch settings.
- Favorites.

> **Important**
>
> USMT doesn't migrate the Start menu layout. To migrate a user's Start menu, settings must be exported and then imported using the Windows PowerShell cmdlets `Export-StartLayout` and **Import-StartLayout**. For more information, see [USMT common issues](#).

- **Folders from the All Users and Public profiles.** When the `MigUser.xml` file is specified, USMT also migrates the following from the **Public** profile in Windows:
  - Shared Documents
  - Shared Video
  - Shared Music
  - Shared desktop files
  - Shared Pictures
  - Shared Start menu
  - Shared Favorites
- **File types.** When the `MigUser.xml` file is specified, the **ScanState** tool searches the fixed drives, collects, and then migrates files with any of the following file extensions:

  `.accdb`, `.ch3`, `.csv`, `.dif`, `.doc*`, `.dot*`, `.dqy`, `.iqy`, `.mcw`, `.mdb*`, `.mpp`, `.one*`, `.oqy`, `.or6`, `.pot*`, `.ppa`, `.pps*`, `.ppt*`, `.pre`, `.pst`, `.pub`, `.qdf`, `.qel`, `.qph`, `.qsd`, `.rqy`, `.rtf`, `.scd`, `.sh3`, `.slk`, `.txt`, `.vl*`, `.vsd`, `.wk*`, `.wpd`, `.wps`, `.wq1`, `.wri`, `.xl*`, `.xla`, `.xlb`, `.xls*`

> **Note**
>
> The asterisk (`*`) stands for zero or more characters.

> **Note**
>
> The OpenDocument extensions (`*.odt`, `*.odp`, `*.ods`) that Microsoft Office applications can use aren't migrated by default.

- **Access control lists.** USMT migrates access control lists (ACLs) for specified files and folders from computers running Windows. For example, if a file named `File1.txt` that is **read-only** for **User1** and **read/write** for **User2** is migrated, these settings will still apply on the destination computer after the migration.

  > **Important**
  >
  > To migrate ACLs, the directory to migrate must be specified in the `MigUser.xml` file. Using file patterns like *.doc won't migrate a directory. The source ACL information is migrated only when the directory is explicitly specified. For example, `<pattern type="File">c:\test docs</pattern>`.

Operating-system components

USMT migrates operating-system components to a destination computer. The following components are migrated by default using the manifest files:

- Accessibility settings.
- Address book.
- Command-prompt settings.
- Desktop wallpaper. [1]
- EFS files.
- Favorites.
- Folder options.
- Fonts.
- Group membership. USMT migrates users' group settings. To view what groups a user belongs to:

  1. Right-clicking on the Start menu and then selecting **Computer Management**.

2. In the **Computer Management** console, expand **System tools** > **Local Users and Groups** > **Groups**.
3. Inspect the individual groups in the results pane to see what users belong to what groups.

The use of a <**ProfileControl**> section in the `Config.xml` file is required when running an offline migration.

- Microsoft Open Database Connectivity (ODBC) settings.
- Mouse and keyboard settings.
- Network drive mapping.
- Network printer mapping. [1]
- Offline files. [1]
- Phone and modem options. [1]
- RAS connection and phone book (.pbk) files.
- Regional settings. [1]
- Remote Access.
- Taskbar settings. [1]
- User personal certificates (all).
- Windows Mail.
- Windows Media Player. [1]
- Windows Rights Management.

[1] These settings aren't available for an offline migration. For more information, see Offline migration reference.

> **Important**
>
> This list might not be complete. There might be additional components that are migrated.

> **Note**

Some settings, such as fonts, aren't applied by the **LoadState** tool until after the destination computer is restarted. For this reason, restart the destination computer after running the **LoadState** tool.

Supported applications

Even though it isn't required for all applications, it's good practice to install all applications on the destination computer before restoring the user state. Installing applications before migrating settings helps to ensure application installers don't overwrite settings that were migrated.

> **Note**
>
> The versions of installed applications must match on the source and destination computers. USMT doesn't support migrating the settings of an earlier version of an application to a later version, except for Microsoft Office.

> **Note**
>
> USMT only migrates settings that are modified on the source computer. If an application setting isn't modified from the default on the source computer, the setting might not migrate.

When the `MigApp.xml` file is specified, USMT migrates the settings for specific applications defined in the `MigApp.xml` file. Consult the `MigApp.xml` file for applications are supported.

What USMT doesn't migrate

The following items are settings that USMT doesn't migrate. If having a problem that isn't listed here, see Common issues.

Application settings

USMT doesn't migrate the following application settings:

- Settings for Microsoft Store applications.

- Settings from earlier versions of an application. The versions of each application must match on the source and destination computers. USMT doesn't support migrating the settings of an earlier version of an application to a later version, except for Microsoft Office. USMT can migrate from an earlier version of Microsoft Office to a later version.

- Application settings and some operating-system settings when a local account is created. For example, if `/lac` is specified to create a local account on the destination computer, USMT migrates the user data, but doesn't migrate:
  - Some operating system settings - Only some operating-system settings, such as wallpaper and screensaver settings, are migrated.
  - Application settings.

Operating-System settings

USMT doesn't migrate the following operating-system settings.

- Local printers, hardware-related settings, drivers, passwords, application binary files, synchronization files, DLL files, or other executable files.

- Permissions for shared folders. After migration, any folders that were shared on the source computer must be manually re-shared.

- Files and settings migrating between operating systems with different languages. The operating system of the source computer must match the language of the operating system on the destination computer.

- Customized icons for shortcuts might not migrate.

Also note the following items:

- Run USMT from an account with administrative credentials. Otherwise, some data doesn't migrate. When the **ScanState** and **LoadState** tools are run, the tools must be run in Administrator mode from an account with administrative credentials. If USMT isn't run in Administrator mode, only the user profile that is logged on is included in the migration.

- Use the `/localonly` option to exclude the data from removable drives and network drives mapped on the source computer. For more information about what is excluded when `/localonly` is specified, see ScanState syntax.

Start menu layout

USMT doesn't migrate the Start menu layout. To migrate a user's Start menu, settings must be exported and then imported using the Windows PowerShell cmdlets `Export-StartLayout` and **Import-StartLayout**. For more information, see USMT common issues.

User profiles from Active Directory to Microsoft Entra ID

- USMT doesn't support migrating user profiles from Active Directory domain joined devices to Microsoft Entra joined devices.
- USMT doesn't support migrating user profiles from Microsoft Entra joined devices to Active Directory domain joined devices.
- USMT doesn't support migrating user profiles between Microsoft Entra joined devices.
- USMT might work when migrating user profiles between Microsoft Entra hybrid joined devices or between Active Directory domain joined devices and Microsoft Entra hybrid joined devices, but it's not a tested scenario so therefore unsupported.

Related articles

- Plan the migration.

Feedback

Was this page helpful? Yes No

Provide product feedback

Choose a migration store type

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

One of the main considerations for planning the migration is to determine which migration store type best meets the organization's needs. As part of these considerations, determine the following items:

- How much space is required to run the User State Migration Tool (USMT) components on the source and destination computers.
- How much space is needed to create and host the migration store.
- Whether a local share, network share, or storage device should be used.
- Ensure that user date integrity is maintained by encrypting the migration store.

In this section

Expand table

Link	Description
Migration store types overview	Choose the migration store type that works best for the organization's needs and migration scenario.
Estimate migration store size	Estimate the amount of disk space needed for computers in the organization based on information about the organization's infrastructure.
Hard-link migration store	Learn about hard-link migration stores and the scenarios in which they're used.
Migration store encryption	Learn about the using migration store encryption to protect user data integrity during a migration.

Related articles

- Plan the migration
- User State Migration Tool (USMT) how-articles

Feedback

Was this page helpful? Yes No

Provide product feedback

Migration Store Types Overview

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

When a migration is being planned, which migration store type best meets the organization's needs should be determined. As part of these considerations, determine how much space is required to run the User State Migration Tool (USMT) components on the source and destination computers. The space needed to create and host the migration store should also be determined, whether using a local share, network share, or storage device.

Migration store types

This section describes the three migration store types available in USMT.

Uncompressed (UNC)

The uncompressed (UNC) migration store is an uncompressed directory with a mirror image of the folder hierarchy being migrated. Each directory and file retains the same access permissions that it has on the local file system. Windows Explorer can be used to view this migration store type. Settings are stored in a catalog file that also describes how to restore files on the destination computer.

Compressed

The compressed migration store is a single image file that contains all files being migrated and a catalog file. This image file is often encrypted and protected with a password, and can't be navigated with Windows Explorer.

Hard-Link

A hard-link migration store functions as a map that defines how a collection of bits on the hard disk are "wired" into the file system. The USMT hard-link migration store is only used in the PC Refresh scenario. Hard-link migration stores are only used in Refresh scenarios because the hard-link migration store is maintained on the local computer. The hard-link store is maintained on the computer while the old operating system is removed and the new operating system is installed. Using a hard-link migration store saves network bandwidth and minimizes the server use needed to accomplish the migration.

The command-line option `/hardlink` is used to create a hard-link migration store, which functions the same as an uncompressed migration store. Files aren't duplicated on the local computer when user state is captured. They also aren't duplicated when user state is restored. For more information, see Hard-Link Migration Store.

The following flowchart illustrates the procedural differences between a local migration store and a remote migration store. In this example, a hard-link migration store is used for the local store.

Capture user state locally
Local store enabled?
No
Copy every file to be migrated to migration store
Yes
Create a hard link in migration store to every file to be migrated
Capture settings in migration store
Clean system and install new operating system
Restore user state
Local store?
No
Copy every file in migration store to proper location
Yes
Hard link every file in migration store to proper location
Restore settings

Local store vs. remote store

If there's enough space and the user state is being migrated back to the same computer, storing data on a local device is normally the best option to reduce server storage costs and network performance issues. The data can also be stored locally either on a different partition or on a removable device such as a USB flash drive (UFD). Also, the data might be able to be stored on the partition that is being re-imaged if the data can be protected from deletion during the imaging process. One example of an imaging technology that is capable of storing the data on the partition that is being reimaged is Microsoft Configuration Manager. To increase performance, store the data on high-speed drives that use a high-speed network connection. It's also good practice to ensure that the migration is the only task the server is performing.

If there isn't enough local disk space, or if moving the user state to another computer, then the data must be stored remotely such as in one of the following destinations:

- Shared folder.
- Removable media.
- Directly on the destination computer.

For example:

1. Create and share `C:\store` on the destination computer.

2. Run the `ScanState.exe` command on the source computer and save the files and settings to `\\<DestinationComputerName>\store`.

3. Run the `LoadState.exe` command on the destination computer and specify `C:\Store` as the store location.

By doing this process, files don't need to be stored to a server.

> **ⓘ Important**
>
> If possible, have users store their data within their `%UserProfile%\Documents` and `%UserProfile%\Application Data` folders. Having users store their data at these locations reduces the chance of USMT missing critical user data that is located in a directory that USMT isn't configured to check.

The /localonly command-line option

This option should be used to exclude the data from removable drives and network drives mapped on the source computer. For more information about what is excluded when `/LocalOnly` is specified, see ScanState Syntax.

Related articles

- Plan the migration.

Feedback

Was this page helpful? Yes No

Provide product feedback

Estimate migration store size

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

The disk space requirements for a migration are dependent on the size of the migration store and the type of migration. The amount of disk space needed for computers in the organization can be estimated based on information about the organization's infrastructure. Disk space requirements can also be calculated using the **ScanState** tool.

Hard disk space requirements

- **Store**: For non-hard-link migrations, ensure that there's enough available disk space at the location where the store is saved. The store contains the data being migrated. The store can be saved to another partition, an external storage device such as a USB flash drive, or a server. For more information, see Choose a Migration Store Type.

- **Source Computer**: The source computer needs enough available space for the following items:

  - **E250 megabytes (MB) minimum of hard disk space**: Space is needed to support the User State Migration Tool (USMT) operations, for example, growth in the page file. If every volume involved in the migration is formatted as NTFS, 250 MB should be enough space to ensure success for almost every hard-link migration, regardless of the size of the migration. The USMT tool that captures data (ScanState) doesn't create the migration store if 250 MB of disk space isn't available.

  - **Temporary space for USMT to run**: Extra disk space is required for the USMT tools to operate. This disk space requirement doesn't include the minimum 250 MB needed to create the migration store. The amount of temporary space required can be calculated using the **ScanState** tool.

  - **Hard-link migration store**: It isn't necessary to estimate the size of a hard-link migration store. The only case where the hard-link store can be large is when non-NTFS file volumes exist on the system and those volumes contain data being migrated.

- **Destination computer**: The destination computer needs enough available space for the following components:

  - **Operating system**.

- **Applications**.
- **Data being migrated**: Data being migrated includes files and registry information.
- **Temporary space for USMT to run**: Extra disk space is required for the USMT tools to operate. The amount of temporary space required can be calculated using the **ScanState** tool.

Calculate disk space requirements using the ScanState tool

The **ScanState** tool can be used to calculate the disk space requirements for a particular compressed or uncompressed migration. It isn't necessary to estimate the migration store size for a hard-link migration since this method doesn't create a separate migration store. The **ScanState** tool provides disk space requirements for the state of the computer at the time the tool is run. The state of the computer might change during day-to-day use. For this reason, use the calculations as an estimate when planning the migration.

To run the **ScanState** tool on the source computer with USMT installed:

1. Open a command prompt with administrator privileges.
2. Navigate to the USMT tools. For example, enter:

   Windows Command Prompt

   ```
   cd /d "C:\Program Files (x86)\Windows Kits\10.0\Assessment and
   Deployment Kit\User State Migration Tool\<architecture>"
   ```

   where <*architecture*> is x86 or amd64.

3. Run the **ScanState** tool to generate an XML report of the space requirements. At the command prompt, enter:

   Windows Command Prompt

   ```
   ScanState.exe <StorePath> /p:<path to a file>
   ```

   Where:

   - <*StorePath*> is a path to a directory where the migration store is saved.

- *<path to a file>* is the path and filename where the XML report for space requirements is saved.

For example:

Windows Command Prompt

```
ScanState.exe c:\store /p:c:\spaceRequirements.xml
```

Although a migration store isn't created by running this command, the *<StorePath>* is still a required parameter.

The **ScanState** tool also allows estimation of disk space requirements based on a customized migration. For example, the **Documents** folder might need to be migrated to the destination computer. This condition can be specified in a configuration file when the **ScanState** tool is run. For more information, see Customize USMT XML files.

> **Note**
>
> To preserve the functionality of existing applications or scripts that require the previous behavior of USMT, the `/p` option is still available in USMT without having to specify the path to a file. For more information, see Monitoring Options.

The space requirements report provides two elements, <**storeSize**> and <**temporarySpace**>. The <**temporarySpace**> value shows the disk space, in bytes, that USMT uses to operate during the migration but it doesn't include the minimum 250 MB needed to support USMT. The <**storeSize**> value shows the disk space, in bytes, required to host the migration store contents on both the source and destination computers. The following example shows a report generated using `/p:`*<path to a file>*.

XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<PreMigration>
  <storeSize>
    <size clusterSize="4096">11010592768</size>
  </storeSize>
  <temporarySpace>
    <size>58189144</size>
  </temporarySpace>
</PreMigration>
```

Additionally, USMT performs a compliance check for a required minimum of 250 MB of available disk space and doesn't create a store if the compliance check fails.

Estimating migration store size

Determine how much space is needed to store the migrated data. Calculations should be based on the volume of e-mail, personal documents, and system settings for each user. The best way to estimate the required space is to survey several computers to arrive at an average for the size of the store that is needed.

The amount of space that is required in the store varies and depends on the local storage strategies the organization uses. For example, one key element that determines the size of migration data sets is e-mail storage. If e-mail is stored centrally, data sets are smaller. If e-mail is stored locally, such as offline-storage files, data sets are larger. Mobile users typically have larger data sets than workstation users. Tests should be performed and the network inventoried to determine the average data set size in the organization.

> **Note**
>
> A space-estimate file (`Usmtsize.txt`) can be created to estimate the size of the store by using the legacy `/p` command-line option.

When trying to determine how much disk space is needed, consider the following issues:

- **E-mail**: If users deal with a large volume of e-mail or keep e-mail on their local computers instead of on a mail server, the e-mail can take up as much disk space as all other user files combined. Prior to migrating user data, make sure that users who store e-mail locally synchronize their inboxes with their mail server.

- **User documents**: Frequently, all of a user's documents fit into less than 50 MB of space, depending on the types of files involved. This estimate assumes typical office work, such as word-processing documents and spreadsheets. This estimate can vary substantially based on the types of documents that the organization uses. For example, an architectural firm that predominantly uses computer-aided design (CAD) files needs more space than a law firm that primarily uses word-processing documents. Documents that users store on file servers through mechanisms such as Folder Redirection don't need to be migrated, as long as users will have access to these locations after the migration.

- **User system settings**: Five megabytes is adequate space to save the registry settings. This requirement can fluctuate, however, based on the number of applications that are installed. It's rare, however, for the user-specific portion of the registry to exceed 5 MB.

Related articles

- Common migration scenarios.

Feedback

Was this page helpful? Yes No

Provide product feedback

Hard-Link Migration Store

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

A **hard-link migration store** enables an in-place migration to be performed where all user state is maintained on the computer while the old operating system is removed and the new operating system is installed. This functionality is what makes **hard-link migration store** best suited for the computer-refresh scenario. Use of a hard-link migration store for a computer-refresh scenario drastically improves migration performance and significantly reduces hard-disk utilization, reduces deployment costs, and enables entirely new migration scenarios.

When to use a hard-link migration

A hard-link migration store can be used when the planned migration meets both of the following criteria:

- The operating system is being upgraded on existing hardware rather than migrating to new computers.
- The operating system is being upgraded on the same volume of the computer.

A hard-link migration store can't be used if the planned migration includes any of the following tasks:

- Data is being migrated from one computer to a different computer.
- Data is being migrating from one volume on a computer to another volume on the same computer, for example from `C:` to `D:`.
- The disk containing the migration store is being formatted or repartitioned disk either outside of Windows Setup or during Windows Setup.

Understanding a hard-link migration

The hard-link migration store is created using the command-line option, `/hardlink`, and is equivalent to other migration-store types. However, it differs in that hard links are utilized to keep files stored on the source computer during the migration. Keeping the files in place on the source computer eliminates the redundant work of duplicating files. It also enables the performance benefits and reduction in disk utilization that define this scenario.

When a hard link is created, an existing file is given one more path. For instance, a hard link to `c:\file1.txt` can be created called `c:\hard link\myFile.txt`. These two paths relate to the same file. If `c:\file1.txt` is opened, then changes made to the file followed by the file being saved, those changes are seen when `c:\hard link\myFile.txt` is opened. If `c:\file1.txt` is deleted, the file still exists on the computer as `c:\hardlink\myFile.txt`. Both references to the file must be deleted in order to delete the file.

> **Note**
>
> A hard link can only be created for a file on the same volume. If a hard-link migration store is copied to another drive or external device, the files, and not the links, are copied, as in a non-compressed migration-store scenario.

For more information about hard links, see Hard Links and Junctions

In most aspects, a hard-link migration store is identical to an uncompressed migration store. The hard-link migration store is located as specified by the **ScanState.exe** command-line tool. The contents of the store can be viewed by using Windows Explorer. Once created, it can be deleted or copied to another location without changing user state. Restoring a hard-link migration store is similar to restoring any other migration store. However, as with creating the store, the same hard-link functionality is used to keep files in-place.

As a best practice, delete the hard-link migration store after confirming that the files are successfully migrated via the **LoadState** tool. Since **LoadState** creates new paths to the files on the new installation of a Windows operating system, deleting the hard links in the migration store only deletes one path to the files. It doesn't delete the actual files or the paths to them from the new operating system.

> **Important**
>
> Using the `/c` option forces the **LoadState** tool to continue applying files when non-fatal errors occur. If the `/c` option is used, verify that no errors are reported in the logs before deleting the hard-link migration store in order to avoid data loss.

Keeping the hard-link migration store can result in extra disk space being consumed or problems with some applications for the following reasons:

- Applications reporting file-system statistics, for example, space used and free space, might incorrectly report these statistics while the hard-link migration store

is present. The file might be reported twice because of the two paths that reference that file.

- A hard link might lose its connection to the original file. Some applications save changes to a file by creating a temporary file and then renaming the original to a backup filename. The path that wasn't used to open the file in this application continues to refer to the unmodified file. The unmodified file that isn't in use is taking up more disk space. The hard-link migration store should be created just before the migration is performed. Once the store is created, applications shouldn't be used in order to make sure the latest versions of all files are being migrating.
- Editing the file by using different paths simultaneously might result in data corruption.

> **Important**
>
> The read-only file attribute on migrated files is lost when the hard-link migration store is deleted. This is due to a limitation in NTFS file system hard links.

Hard-link migration scenario

For example, an organization decides to deploy the latest supported version of Windows on all of their computers. Each employee keeps the same computer, but the operating system on each computer will be updated.

1. An administrator runs the **ScanState** command-line tool on each computer, specifying the `/hardlink` command-line option. The **ScanState** tool saves the user state to a hard-link migration store on each computer, improving performance by reducing file duplication, except in certain specific instances.

   > **Note**
   >
   > As a best practice, Microsoft recommends not to create the hard-link migration store until just before the migration is performed in order to migrate the latest versions of files. Software applications shouldn't be used on the computer after creating the migration store until files finish migrating with **LoadState**.

2. On each computer, an administrator installs the organization's standard operating environment (SOE), which includes the latest supported version of Windows and other applications the organization currently uses.

3. An administrator runs the **LoadState** command-line tool on each computer. The **LoadState** tool restores user state back on each computer.

> **Note**
>
> During the update of a domain-joined computer, the profiles of users whose SID cannot be resolved will not be migrated. When using a hard-link migration store, it could cause a data loss.

Hard-link migration store details

This section provides details about hard-link migration stores.

Hard disk space

The `/hardlink` command-line option proceeds with creating the migration store only if there are 250 megabytes (MB) of free space on the hard disk. If every volume involved in the migration is formatted as NTFS, 250 MB should be enough space to ensure success for almost every hard-link migration, regardless on the size of the migration.

Hard-link store size estimation

It isn't necessary to estimate the size of a hard-link migration store since a hard-link migration store on an NTFS volume is relatively small and require much less incremental space than other store options. Estimating the size of a migration store is only useful in scenarios where the migration store is large. The only case where the local store can be large with hard-link migrations is:

- A non-NTFS file system exists on the system.
- The non-NTFS files system contains data that needs to be migrated.

Since NTFS is the default file system format for all currently supported versions of Windows, this situation is unusual.

Migration store path on multiple volumes

Separate hard-link migration stores are created on each NTFS volume that contain data being migrated. In this scenario, the primary migration-store location is specified on the command line, and should be the operating-system volume. Migration stores with identical names and directory names are created on every volume containing data being migrated. For example:

Windows Command Prompt

```
ScanState.exe /hardlink c:\USMTMIG […]
```

Running this command on a system that contains the operating system on the C: drive and the user data on the D: drive generates migration stores in the following locations, assuming that both drives are NTFS:

`C:\USMTMIG\`

`D:\USMTMIG\`

The drive specified on the command line for the hard-link migration store is important, because it defines where the **master migration store** should be placed. The **master migration store** is the location where data migrating from non-NTFS volumes is stored. This volume must have enough space to contain all of the data that comes from non-NTFS volumes. As in other scenarios, if a migration store already exists at the specified path, the `/o` option must be used to overwrite the existing data in the store.

Location modifications

Location modifications that redirect migrated content from one volume to a different volume have an adverse effect on the performance of a hard-link migration. Performance is affected because the migrating data that must cross system volumes can't remain in the hard-link migration store. They must be copied across the system volumes.

Migrating Encrypting File System (EFS) certificates and files

To migrate Encrypting File System (EFS) files to a new installation of an operating system on the same volume of the computer, specify the `/efs:hardlink` option in the `ScanState.exe` command-line syntax.

If the EFS files are being restored to a different partition, the `/efs:copyraw` option should be used instead of the `/efs:hardlink` option. Hard links can only be created for files on the same volume. Moving the files to another partition during the migration requires a copy of the files to be created on the new partition. The `/efs:copyraw` option copies the files to the new partition in encrypted format.

For more information, see Migrate EFS files and certificates and Encrypted file options.

Migrating locked files with the hard-link migration store

When an application or the operating system has a lock on a file, the file is handled differently when using a hard-link migration store.

Operating system locked files can't remain in place and must be copied into the hard-link migration store. As a result, selecting many operating-system files for migration significantly reduces performance during a hard-link migration. As a best practice, Microsoft recommends not migrating any files out of the `\Windows` directory, which minimizes performance-related issues.

Application locked files are treated the same in hard-link migrations as in other scenarios when the volume shadow-copy service isn't being utilized. The volume shadow-copy service can't be used with hard-link migrations. However, by modifying the new **<HardLinkStoreControl>** section in the `Config.xml` file, it's possible to enable the migration of files locked by an application.

> **Important**
>
> There are some scenarios in which modifying the **<HardLinkStoreControl>** section in the `Config.xml` file makes it more difficult to delete a hard-link migration store. In these scenarios, `UsmtUtils.exe` must be used to schedule the migration store for deletion on the next restart.

XML elements in the Config.xml file

A new section in the `Config.xml` file allows optional configuration of some of the hard-link migration behavior introduced with the `/HardLink` option.

Expand table

Element	Description
<Policies>	This element contains elements that describe the policies that USMT follows while creating a migration store.
<HardLinkStoreControl>	This element contains elements that describe how to handle files during the creation of a hard link migration store.
<fileLocked>	This element contains elements that describe how to handle files that are locked for editing.
<createHardLink>	This element defines a standard MigXML pattern that describes file paths where hard links should be created, even if the file is locked for editing by another application. Syntax: `<createHardLink>` [pattern] `</createHardLink>`
<errorHardLink>	This element defines a standard MigXML pattern that describes file paths where hard links shouldn't be created, if the file is locked for editing by another application. `<errorHardLink>` [pattern] `</errorHardLink>`

> **Important**
>
> The `/nocompress` option must be used with the `/HardLink` option.

The following XML sample specifies that files locked by an application under the `\Users` directory can remain in place during the migration. It also specifies that locked files that aren't located in the `\Users` directory should result in the **File in Use** error. It's important to exercise caution when specifying the paths using the **`<createhardlink>`** tag in order to minimize scenarios that make the hard-link migration store more difficult to delete.

XML

```xml
<Policies>
    <HardLinkStoreControl>
          <fileLocked>
            <createHardLink>c:\Users\* [*]</createHardLink>
            <errorHardLink>C:\* [*]</errorHardLink>
          </fileLocked>
    </HardLinkStoreControl>
</Policies>
```

Related articles

- Plan the migration.

Feedback

Was this page helpful? Yes No

Provide product feedback

Migration store encryption

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

This article discusses User State Migration Tool (USMT) options for migration store encryption to protect the integrity of user data during a migration.

USMT encryption options

USMT enables support for stronger encryption algorithms, called Advanced Encryption Standard (AES), in several bit-level options. AES is a National Institute of Standards and Technology (NIST) specification for the encryption of electronic data.

The chosen encryption algorithm must be specified for both the `ScanState.exe` and the `LoadState.exe` commands, so that these commands can create or read the store during encryption and decryption. The new encryption algorithms can be specified on the `ScanState.exe` and the `LoadState.exe` command lines by using the `/encrypt`:*encryption_strength* and the `/decrypt`:*encryption_strength* command-line options. All of the encryption application programming interfaces (APIs) used by USMT are available in currently supported versions of Windows. However, export restrictions might limit the set of algorithms that are available to computers in certain locales. The `UsmtUtils.exe` file can be used to determine which encryption algorithms are available to the computers' locales before the migration begins.

The following table describes the command-line encryption options in USMT.

Expand table

Component	Option	Description
ScanState	**/encrypt**<*AES, AES_128, AES_192, AES_256, 3DES, 3DES_112*>	This option and argument specify that the migration store is encrypted and which algorithm to use. When the algorithm argument isn't provided, the **ScanState** tool employs the **3DES** algorithm.
LoadState	**/decrypt**<*AES, AES_128, AES_192, AES_256, 3DES, 3DES_112*>	This option and argument specify that the store must be decrypted and which algorithm to use. When the algorithm argument isn't provided, the **LoadState** tool employs the **3DES** algorithm.

Important

Some encryption algorithms might not be available on some systems. Which algorithms are available can be verified by running the `UsmtUtils.exe` command with the `/ec` option. For more information, see **UsmtUtils syntax**.

Related articles

- Plan the migration.

Feedback

Was this page helpful? Yes No

Provide product feedback

Determine what to migrate

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

By default, User State Migration Tool (USMT) migrates the items listed in What does USMT migrate?, depending on the migration **.xml** files that are specified. These default settings are often enough for a basic migration.

However, when considering what settings to migrate, also consider:

- What settings the user can configure, if any.
- What settings should be standardized.

Many organizations use their migration as an opportunity to create and begin enforcing a better-managed environment. Some of the settings that users can configure on unmanaged computers prior to the migration can be locked on the new, managed computers. For example, standard wallpaper and desktop configuration are some of the items that can be standardized.

To reduce complexity and increase standardization, the organization should consider creating a *standard operating environment (SOE)*. An SOE is a combination of hardware and software that is distributed to all users. Creating an SOE means selecting:

- A baseline for all computers, including standard hardware drivers.
- Core operating system features.
- Core productivity applications, especially if they are under volume licensing.
- Core utilities.
- A standard set of security features, as outlined in the organization's corporate policy.

Using an SOE can vastly simplify the migration and reduce overall deployment challenges.

In this section

Expand table

Link	Description
Identify users	Use command-line options to specify which users to migrate and how they should be migrated.
Identify applications	Determine which applications need to be migrated and prepare a list

Link	Description
settings	of application settings to be migrated.
Identify operating system settings	Use migration to create a new standard environment on each of the destination computers.
Identify file types, files, and folders	For the following items that need to be migrated: • File types. • Files. • Folders. • Settings. determine where these items might be located. For example: • Standard default OS locations. • Organization-specified locations. • Non-standard locations.

Related articles

- What does USMT migrate?.

Feedback

Was this page helpful? Yes No

Provide product feedback

Identify users

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

It's important to carefully consider and plan how users are migrated. By default, User State Migration Tool (USMT) migrates all users. Which users to include must be specified by using the command line. Users can't be specified in the **.xml** files. For instructions on how to migrate users, see Migrate user accounts.

Migrating local accounts

Before migrating local accounts, be aware of the following items:

- **Local accounts that aren't on the destination computer must be explicitly specified if they should be migrated.** If migrating local accounts and the local account doesn't exist on the destination computer, the `/lac` option must be specified when using the `LoadState.exe` command. If the `/lac` option isn't specified, no local user accounts are migrated.
- **Consider whether to enable user accounts that are new to the destination computer.** The `/lae` option enables the account that was created with the `/lac` option. However, if a disabled local account is created by using only the `/lac` option, a local administrator must enable the account on the destination computer.
- **Be careful when specifying a password for local accounts.** If the local account is created with a blank password, anyone could sign in that account on the destination computer. If the local account is created with a password, the password is available to anyone with access to the USMT command-line tools.

> **Note**
>
> If there are multiple users on a computer, and a password is specified with the `/lac` option, all migrated users have the same password.

Migrating domain accounts

The source and destination computers don't need to be connected to the domain for domain user profiles to be migrated.

Command-line options

USMT provides several options to migrate multiple users on a single computer. The following command-line options specify which users to migrate.

- **Specifying users.** Which users to migrate can be specified with the `/all`, `/ui`, `/uel`, and `/ue` options with both the **ScanState** and **LoadState** command-line tools.

  > **Important**
  >
  > The `/uel` option excludes users based on the **LastModified** date of the `Ntuser.dat` file. The `/uel` option isn't valid in offline migrations.

- **Moving users to another domain.** User accounts can be moved to another domain using the `/md` option with the **LoadState** command-line tool.
- **Creating local accounts.** Local accounts can be created and enabled using the `/lac` and `/lae` options with the **LoadState** command-line tool.
- **Renaming user accounts.** User accounts can be renamed using the `/mu` option.

  > **Note**
  >
  > By default, if a user name isn't specified in any of the command-line options, the user is migrated.

Related articles

- Determine what to migrate.
- ScanState syntax.
- LoadState syntax.

Feedback

Was this page helpful? Yes No

Provide product feedback

Identify applications settings

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

Which applications and settings need to be migrated should be identified when planning a migration. For more information about how to create a custom **.xml** file to migrate the settings of another application, see Customize USMT XML files.

Applications

First, create and prioritize a list of applications that need to be migrated. It might be helpful to review the application lists and decide which applications need to be redeployed and which applications need to be retired. Often, how the application is used and how complex the application is determines the priority of what applications are migrated.

Next, identify an application owner to be in charge of each application. Application ownership identification is necessary because the developers aren't be experts on all of the applications in the organization. The application owner should have the most experience with an application. The application owner provides insight into how the organization installs, configures, and uses the application.

Application settings

Next, determine and locate the application settings to be migrated. Much of the information that is needed for this step can be acquired when testing the new applications for compatibility with the new operating system.

After completing the list of applications to be migrated, review the list, and work with each application owner on a list of settings to be migrated. For each setting, determine whether it needs to be migrated or if the default settings are adequate. Then, determine where the setting is located, for example, in the registry or in an **.ini** file. Next, consider the following questions to determine what needs to be done to migrate the setting successfully:

- Is the destination version of the application newer than the source version?
- Do these settings work with the new version?
- Do the settings need to be moved or altered?

- Can the first-run process force the application to appear as if it ran already? If so, does this work correctly, or does it break the application?

After answering these questions, create a custom **.xml** file to migrate settings. Work with the application owner to develop test cases and to determine the file types that need to be migrated for the application.

Locating where settings are stored

See Migrate application settings and follow the directions.

Related articles

- Determine what to migrate.

Feedback

Was this page helpful? Yes No

Provide product feedback

Identify operating system settings

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

When the migration is being planned, which operating system settings need to be migrated should be identified. Additionally, to what extent a new standard environment should be created on each of the computers should also be identified. User State Migration Tool (USMT) enables migrating select settings and keep the default values for all others. The operating system settings include the following parameters:

- **Appearance**

  The appearance factor includes items such as wallpaper, colors, sounds, and the location of the taskbar.

- **Action**

  The action factor includes items such as:
  - The key-repeat rate.
  - Whether double-clicking a folder opens it in a new window or the same window.
  - Whether single-clicking or double-clicking an item opens it.

- **Internet**

  The Internet factor includes the settings needed to connect to the Internet and controls how the browser operates. The settings include items such as the home page URL, favorites, bookmarks, cookies, security settings, and proxy settings. These settings might not be supported in all browsers.

- **Mail**

  The mail factor includes the information needed to connect the mail server, the signature file, views, mail rules, local mail, and contacts. These settings might not be supported in all email applications.

To help determine which settings to migrate, consider any previous migration experiences and the results of any conducted surveys and tests. Also consider the number of help-desk calls related to operating-system settings from the past, and are able to handle in the future. Also decide how much of the new operating-system functionality needs to be taken advantage of.

Settings that should be migrated include:

- Settings that allow users need to get their jobs done.

- Settings that make the work environment comfortable.
- Settings that will reduce help-desk calls after the migration.

Although it's easy to dismiss migrating user preferences, the factor should be considered of users spending time restoring items such as:

- Wallpaper.
- Screen savers.
- Other customizable user-interface features.

Most users don't remember how these settings were applied. Although these items aren't critical to migration success, migrating these items increases user productivity and overall satisfaction of the migration process.

> **Note**
>
> For more information about how to change the operating-system settings that are migrated, see User State Migration Tool (USMT) how-to articles.

For information about the operating-system settings that USMT migrates, see What does USMT migrate?

Related articles

- Determine What to Migrate.

Feedback

Was this page helpful? Yes No

Provide product feedback

Identify file types, files, and folders

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

When a migration is planned and `MigDocs.xml` isn't being used, the file types, files, folders, and settings that need to be migrated should be identified. First, the standard file locations on each computer, such as the **Documents** folder, `C:\Data` , and organization-specified locations, such as `\\EngineeringDrafts`, should be determined. Next, non-standard locations should be determined and located. For non-standard locations, consider the following items:

- **File types**: Consider which file types need to be included and excluded from the migration. This list can be created based on common applications used in the organization. Applications normally use specific file name extensions. For example, Microsoft Office Word primarily uses `.doc`, `.docx` and `.dotx` file name extension. However, it also uses other file types, such as templates (`.dot` files), on a less frequent basis.

- **Excluded locations**: Consider the locations on the computer that should be excluded from the migration (for example, `%WINDIR%` and **Program Files**).

- **New locations**: Decide where files should be migrated to on the destination computer, such as the **Documents** folder, a designated folder, or a folder matching the files' name and location on the source computer. For example, shared data might exist on the source machine or documents outside the user profiles on the source system might need to be cleaned up. Identify any data that needs to be redirected to a new location in the Apply phase. Redirection can be accomplished with location modify rules.

Once which files and file types that the end users work with regularly is verified, locate the files and file types. Files might be saved to a single folder or scattered across a drive. A good starting point for finding files types to include is to look at the registered file types on the computer.

To find the registered file types on a computer running a currently supported version of Windows:

1. Right-click the **Start Menu** and select **Settings**.

2. When the **Settings** window opens, select **Apps**.

3. Select **Default apps**.

4. Scroll down and then select **Choose defaults by file type** or **Choose default apps by file type**.

5. In the window that opens, the registered file types are displayed.

For more information about how to change the file types, files, and folders that are migrated when the `MigUser.xml` file is specified, see User State Migration Tool (USMT) how-to articles.

Related articles

- Determine what to migrate.

Feedback

Was this page helpful? Yes No

Provide product feedback

Test the migration

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

Always test the migration plan in a controlled laboratory setting before deploying it to the entire organization. In the test environment, at least one computer is needed for each type of operating system from which data is being migrated.

Once the entire migration process is tested on a single computer running each of the organization source operating systems, conduct a pilot migration with a small group of users. After migrating a few typical user states to the intermediate store, note the space required and adjust the initial calculations accordingly. For details about estimating the space needed for the migration, see Estimate migration store size. Registry-setting and file-location information might need to be adjusted in the migration-rule files. If changes are made, test the migration again and verify that all data and settings migrated as expected. A pilot migration also gives the opportunity to test the space estimates for the intermediate store.

If the test migration encounters any errors, examine the **ScanState** and **LoadState** logs to obtain the exact User State Migration Tool (USMT) return code and associated error messages or Windows application programming interface (API) error message. For more information about USMT return codes and error messages, see Return codes. More information can be obtained about any listed **Windows** system error codes by typing in the following at a command prompt window:

Windows Command Prompt

```
net.exe helpmsg <error_number>
```

where <*error_number*> is the error code number generated by the error message. For more information about System Error Codes, see System Error Codes (0-499).

In most cases, the **ScanState** and **LoadState** logs indicate why a USMT migration is failing. Microsoft recommends using the `/v:5` option when testing the migration. This verbosity level can be adjusted in a production migration. Reducing the verbosity level might make it more difficult to diagnose failures that are encountered during production migrations. Higher verbosity levels can be used if the log files need to output to go to a debugger.

ⓘ **Note**

Running the **ScanState** and **LoadState** tools with the `/v:5` option creates a detailed log file. Although this option makes the log file large, it's helpful in determining where migration errors occurred.

After the pilot migration is verified that it successfully migrated the specified files and settings, USMT is ready to be used in the environment to migrate data. For example, using USMT with Microsoft Configuration Manager. For more information, see [Manage user state in Configuration Manager]/(mem/configmgr/osd/get-started/manage-user-state).

> **Note**
>
> For testing purposes, an uncompressed store using the `/hardlink /nocompress` option can be created. When compression is disabled, the **ScanState** tool saves the files and settings to a hidden folder named **File** at `<StorePath>\USMT`. The uncompressed store can be used to view what USMT stored or to troubleshoot a problem. An antivirus utility can also be run against the files. Additionally, the following items can be used to troubleshoot problems with the migration:
>
> - The `/listfiles` command-line option.
> - The diagnostic log that lists the files that were gathered.

Related articles

- Plan the migration.
- Log files.

Feedback

Was this page helpful? Yes No

Provide product feedback

User State Migration Tool (USMT) command-line syntax

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

The User State Migration Tool (USMT) migrates user files and settings during large deployments of Windows. To improve and simplify the migration process, USMT captures desktop, network, and application settings in addition to a user's files. USMT then migrates these items to a new Windows installation.

In this Section

Expand table

Link	Description
ScanState syntax	Lists the command-line options for using the **ScanState** tool.
LoadState syntax	Lists the command-line options for using the **LoadState** tool.
UsmtUtils syntax	Lists the command-line options for using the UsmtUtils tool.

Feedback

Was this page helpful? Yes No

Provide product feedback

ScanState syntax

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

The `ScanState.exe` command is used with the User State Migration Tool (USMT) to scan the source computer, collect the files and settings, and create a store. This article discusses the `ScanState.exe` command syntax and the options available with it.

Before beginning

Before running the `ScanState.exe` command, note the items:

- To ensure that all operating system settings migrate, in run the `ScanState.exe` commands in administrator mode from an account with administrative credentials.
- If the migration store is encrypted, an encryption key or a path to a file containing the encryption key is required. Be sure to make note of the key or the key file location, because this information isn't kept anywhere in the migration store. This information is needed when the `LoadState.exe` command is run to decrypt the migration store, or if the recovery utility needs to be used. An incorrect or missing key or key file results in an error message.
- For information about software requirements for running the `ScanState.exe` command, see USMT requirements.
- Unless otherwise noted, use each option only once when running a tool on the command line.
- Domain accounts can be gathered without the source computer having domain controller access. This functionality is available without any extra configuration.
- The Incompatible command-line options table lists which options can be used together and which command-line options are incompatible.
- The directory location where the migration store is saved is excluded from the scan. For example, if the migration store is saved to the root of the D drive, the D drive and all of its subdirectories is excluded from the scan.

Syntax

This section explains the syntax and usage of the command-line options available when using the `ScanState.exe` command. The options can be specified in any order. If the

option contains a parameter, either a colon or a space separator can be used.

The `ScanState.exe` command's syntax is:

> ScanState.exe [*StorePath*] [/apps] [/ppkg:*FileName*] [/i:[*Path*\]*FileName*] [/o] [/v:*VerbosityLevel*] [/nocompress] [/localonly] [/encrypt /key:*KeyString*|/keyfile: [Path\]*FileName*] [/l:[*Path*\]*FileName*] [/progress:[*Path*\]*FileName*] [/r:*TimesToRetry*] [/w:*SecondsBeforeRetry*] [/c] [/p] [/all] [/ui: [*DomainName*|*ComputerName*\]*UserName*] [/ue: [*DomainName*|*ComputerName*\]*UserName*] [/uel:*NumberOfDays*|*YYYY/MM/DD*|0] [/efs:abort|skip|decryptcopy|copyraw] [/genconfig:[*Path*\]*FileName*[/config: [*Path*\]*FileName*] [/?|help]

For example, to create a `Config.xml` file in the current directory, use:

Windows Command Prompt

```
ScanState.exe /i:MigApp.xml /i:MigDocs.xml /genconfig:Config.xml /v:13
```

To create an encrypted store using the `Config.xml` file and the default migration **.xml** files, use:

`ScanState.exe \\server\share\migration\mystore /i:MigApp.xml /i:MigDocs.xml /o /config:Config.xml /v:13 /encrypt /key:"mykey"`

Storage options

Expand table

Command-Line Option	Description
StorePath	Indicates a folder where files and settings are saved. *StorePath* can't be `C:\`. The *StorePath* option must be specified in the `ScanState.exe` command, except when using the `/genconfig` option. More than one *StorePath* location can't be specified.
/apps	Scans the image for apps and includes them and their associated registry settings.
/ppkg [<*FileName*>]	Exports to a specific file location.
/o	Required to overwrite any existing data in the migration store or `Config.xml` file. If not specified, the `ScanState.exe` command fails if the

Command-Line Option	Description
	migration store already contains data. This option can't be used more than once on a command line.
/vsc	This option enables the volume shadow-copy service to migrate files that are locked or in use. This command-line option eliminates most file-locking errors that are typically encountered by the **<ErrorControl>** section. This option is only used with the **ScanState** executable file and can't be combined with the `/hardlink` option.
/hardlink	Enables the creation of a hard-link migration store at the specified location. The `/nocompress` option must be specified with the `/hardlink` option.
/encrypt [{**/key:** <*KeyString*> | **/keyfile**: <*file*>]}	Encrypts the store with the specified key. Encryption is disabled by default. With this option, the encryption key needs to be specified in one of the following ways: • `/key`: *KeyString* specifies the encryption key. If there's a space in *KeyString*, *KeyString* needs to be surrounded with quotation marks (`"`). • `/keyfile`: *FilePathAndName* specifies a text (`.txt`) file that contains the encryption key. *KeyString* is recommended to be at least eight characters long, but it can't exceed 256 characters. The `/key` and `/keyfile` options can't be used on the same command line. The `/encrypt` and `/nocompress` options can't be used on the same command line. **Important** Use caution when using the `/key` or `keyfile` options. For example, anyone who has access to scripts that run the `ScanState.exe` command with these options also have access to the encryption key. The following example shows the `ScanState.exe` command and the `/key` option: `ScanState.exe /i:MigDocs.xml /i:MigApp.xml \server\share\migration\mystore /encrypt /key:mykey`
/encrypt: <*EncryptionStrength*>	The `/encrypt` option accepts a command-line parameter to define the encryption strength to be used for encryption of the migration store. For more information about supported encryption algorithms, see Migration Store Encryption.

Command-Line Option	Description
/nocompress	Disables compression of data and saves the files to a hidden folder named "File" at *StorePath*\USMT. Compression is enabled by default. Combining the `/nocompress` option with the `/hardlink` option generates a hard-link migration store. The uncompressed store can be used to view what USMT stored, troubleshoot a problem, or run an antivirus utility against the files. This option should only be used in testing environments. Microsoft recommends using a compressed store during production migrations, unless combining the `/nocompress` option with the `/hardlink` option. The `/nocompress` and `/encrypt` options can't be used together in one statement on the command line. However, if an uncompressed store is migrated, the `LoadState.exe` command migrates each file directly from the store to the correct location on the destination computer without a temporary location. For example: `ScanState.exe /i:MigDocs.xml /i:MigApp.xml \server\share\migration\mystore /nocompress`

Run the ScanState command on an offline Windows system

The `ScanState.exe` command can be run in Windows Preinstallation Environment (WinPE). In addition, USMT supports migrations from previous installations of Windows contained in **Windows.old** directories. The offline directory can be a Windows directory when the `ScanState.exe` command is run in WinPE or a **Windows.old** directory when the `ScanState.exe` command is run in Windows.

There are several benefits to running the `ScanState.exe` command on an offline Windows image, including:

- **Improved performance.**

  Because WinPE is a thin operating system, there are fewer running services. In this environment, the `ScanState.exe` command has more access to the local hardware resources, enabling **ScanState** to perform migration operations more quickly.

- **Simplified end to end deployment process.**

Migrating data from **Windows.old** simplifies the end-to-end deployment process by enabling the migration process to occur after the new operating system is installed.

- **Improved success of migration.**

  The migration success rate is increased because:
  - Files aren't locked for editing while offline.
  - WinPE provides administrator access to files in the offline Windows file system, eliminating the need for administrator-level access to the online system.

- **Ability to recover an from a computer that doesn't boot.**

  It might be possible to recover and migrate data from a computer that doesn't boot.

Offline migration options

Expand table

Command-Line Option	Definition
/offline: *"path to an Offline.xml file"*	This option is used to define a path to an offline **.xml** file that might specify other offline migration options. For example, an offline Windows directory or any domain or folder redirection required in the migration.
/offlinewindir: *"path to a Windows directory"*	This option specifies the offline Windows directory that the `ScanState.exe` command gathers user state from. The offline directory can be **Windows.old** when the `ScanState.exe` command is run in Windows or a Windows directory when the `ScanState.exe` command is run in WinPE.
/offlinewinold: *"Windows.old directory"*	This command-line option enables the offline migration mode and starts the migration from the location specified. This option is only intended to be used in **Windows.old** migration scenarios, where the migration is occurring from a **Windows.old** directory.

Migration rule options

USMT provides the following options to specify what files to migrate.

Expand table

Command-Line Option	Description
/i: [*Path*]*FileName*	**(include)** Specifies an **.xml** file that contains rules that define what user, application, or system state to migrate. This option can be specified multiple times to include all of the **.xml** files (`MigApp.xml`, `MigDocs.xml`, and any custom **.xml** files that are created). *Path* can be either a relative or full path. If the *Path* variable isn't specified, then *FileName* must be located in the current directory. For more information about which files to specify, see the "XML Files" section of the Frequently asked questions article.
/genconfig: [*Path*]*FileName*	(Generate **Config.xml**) Generates the optional `Config.xml` file, but doesn't create a migration store. To ensure that this file contains everything that needs to be migrated, create this file on a source computer that contains all of the: • components • applications • settings • present on the destination computers. In addition, the other migration **.xml** files should be specified, using the **/i** option, when this option is specified. After this file is created, it can be used with the `ScanState.exe` command using the **/config** option. The only options that can be specified with this option are the `/i`, `/v`, and `/l` options. *StorePath* can't be specified, because the `/genconfig` option doesn't create a store. *Path* can be either a relative or full path. If the *Path* variable isn't specified, then *FileName* is created in the current directory. Examples: • The following example creates a `Config.xml` file in the current directory: `ScanState.exe /i:MigApp.xml /i:MigDocs.xml /genconfig:Config.xml /v:13`
/config: [*Path*]*FileName*	Specifies the `Config.xml` file that the `ScanState.exe` command should use to create the store. This option can't be used more than once on the command line. *Path* can be either a relative or full path. If the *Path* variable isn't specified, then *FileName* must be located in the current directory. The following example creates a store using the `Config.xml` file, `MigDocs.xml`, and `MigApp.xml` files: `ScanState.exe \server\share\migration\mystore /config:Config.xml`

Command-Line Option	Description

	`/i:MigDocs.xml /i:MigApp.xml /v:13 /l:ScanState.log`
	
	The following example migrates the files and settings to the destination computer using the `Config.xml`, `MigDocs.xml`, and `MigApp.xml` files:
	`LoadState.exe \server\share\migration\mystore /config:Config.xml /i:MigDocs.xml /i:MigApp.xml /v:13 /l:LoadState.log`

/auto: *path to script files*	This option enables specifying the location of the default **.xml** files. If no path is specified, USMT references the directory where the USMT binaries are located. The `/auto` option has the same effect as using the following options: `/i:MigDocs.xml /i:MigApp.xml /v:5`.
/genmigxml: *path to a file*	This option specifies that the `ScanState.exe` command should use the document finder to create and export an **.xml** file that defines how to migrate all of the files on the computer on which the `ScanState.exe` command is running.
/localonly	Migrates only files that are stored on the local computer, regardless of the rules in the **.xml** files that are specified on the command line. This option should be used to exclude the data from removable drives on the source computer and when there are network drives mapped on the source computer. Examples of removable drives include USB flash drives (UFDs) and some external hard drives. If the `/localonly` option isn't specified, then the `ScanState.exe` command copies files from these removable or network drives into the store. `/localonly` excludes anything that isn't considered a fixed drive by the OS. In some cases, large external hard drives are considered fixed drives. These drives can be explicitly excluded from migration by using a custom **.xml** file. For more information about how to exclude all files on a specific drive, see Exclude files and settings. The `/localonly` command-line option includes or excludes data in the migration as identified in the following storage locations: • **Removable drives such as a USB flash drive** - Excluded • **Network drives** - Excluded • **Fixed drives** - Included

Monitoring options

USMT provides several options that can be used to analyze problems that occur during migration.

> **Note**
>
> The **ScanState** log is created by default, but the name and location of the log can be specified with the **/l** option.

Expand table

Command-Line Option	Description
/listfiles:<FileName>	The `/listfiles` command-line option can be used with the `ScanState.exe` command to generate a text file that lists all of the files included in the migration.
/l:[*Path*]*FileName*	Specifies the location and name of the **ScanState** log. The log files can't be stored in *StorePath*. *Path* can be either a relative or full path. If the *Path* variable isn't specified, then the log is created in the current directory. The `/v` option can be used to adjust the amount of output. If the `ScanState.exe` command is run from a shared network resource, the `/l` option must be specified, or USMT fails with the following error: ***USMT was unable to create the log file(s)*** To fix this issue, make sure to specify the `/l` option when running `ScanState.exe` from a shared network resource.
/v:<*VerbosityLevel*>	**(Verbosity)** Enables verbose output in the **ScanState** log file. The default value is 0. The *VerbosityLevel* can be set to one of the following levels: • **0** - Only the default errors and warnings are enabled. • **1** - Enables verbose output. • **4** - Enables error and status output. • **5** - Enables verbose and status output. • **8** - Enables error output to a debugger. • **9** - Enables verbose output to a debugger. • **12** - Enables error and status output to a debugger. • **13** - Enables verbose, status, and debugger output. For example:

Command-Line Option	Description
	`ScanState.exe \server\share\migration\mystore /v:13 /i:MigDocs.xml /i:MigApp.xml`
/progress: [*Path*]*FileName*	Creates the optional progress log. The log files can't be stored in *StorePath*. *Path* can be either a relative or full path. If the *Path* variable isn't specified, then *FileName* is created in the current directory. For example: `ScanState.exe /i:MigApp.xml /i:MigDocs.xml \server\share\migration\mystore /progress:Progress.log /l:scanlog.log`
/c	When this option is specified, the `ScanState.exe` command continues to run, even if non-fatal errors occur. Any files or settings that cause an error are logged in the progress log. For example, if there's a large file that doesn't fit in the store, the `ScanState.exe` command logs an error and continue with the migration. In addition, if a file is open or in use by an application, USMT might not be able to migrate the file and logs an error. Without the `/c` option, the `ScanState.exe` command exits on the first error. The <**ErrorControl**> section in the `Config.xml` file can be used to specify which file or registry read/write errors can be safely ignored and which might cause the migration to fail. This advantage in the `Config.xml` file enables the `/c` command-line option to safely skip all input/output (I/O) errors in the environment. In addition, the /`genconfig` option now generates a sample <**ErrorControl**> section that is enabled by specifying error messages and desired behaviors in the `Config.xml` file.
/r:<*TimesToRetry*>	**(Retry)** Specifies the number of times to retry when an error occurs while saving the user state to a server. The default is three times. This option is useful in environments where network connectivity isn't reliable. When the user state is stored, the `/r` option can't recover data that is lost due to a network-hardware failure, such as a faulty or disconnected network cable, or when a virtual private network (VPN) connection fails. The retry option is intended for large, busy networks where connectivity is satisfactory, but communication latency is a problem.
/w: <*SecondsBeforeRetry*>	**(Wait)** Specifies the time to wait, in seconds, before retrying a network file operation. The default is 1 second.

Command-Line Option	Description
/p:<*pathToFile*>	When the `ScanState.exe` command runs, it creates an **.xml** file in the path specified. This **.xml** file includes improved space estimations for the migration store. The following example shows how to create this **.xml** file: `ScanState.exe C:\MigrationLocation [additional parameters] /p:"C:\MigrationStoreSize.xml"` For more information, see Estimate Migration Store Size. To preserve the functionality of existing applications or scripts that require the previous behavior of USMT, the `/p` option can be used, without specifying "*pathtoafile*", in USMT. If only the `/p` option is specified, the storage space estimations are created in the same manner as with USMT 3.x releases.
/? or **/help**	Displays Help at the command line.

User options

By default, all users are migrated. The only way to specify which users to include and exclude is by using the following options. Users can't be excluded in the migration **.xml** files or using the `Config.xml` file. For more information, see Identify users and Migrate user accounts.

Expand table

Command-Line Option	Description
/all	Migrates all of the users on the computer. USMT migrates all user accounts on the computer, unless an account is specifically excluded with either the `/ue` or `/uel` options. For this reason, this option doesn't need to be specified on the command line. However, if the `/all` option is specified, the `/ui`, `/ue` or `/uel` options can't also be specified.
/ui:<*DomainName*>\<*UserName*> or **/ui:**<*ComputerName*>\<*LocalUserName*>	**(User include)** Migrates the specified users. By default, all users are included in the migration. Therefore, this option is helpful only when used with the `/ue` or `/uel` options. Multiple `/ui` options can be specified, but the `/ui` option can't be used with the `/all` option. *DomainName* and *UserName*

Command-Line Option	Description

	can contain the asterisk (`*`) wildcard character. When a user name that contains spaces is specified, it needs to be surrounded with quotation marks (`"`).
	
	Note
	If a user is specified for inclusion with the `/ui` option and also specified to be excluded with either the `/ue` or `/uel` options, the user is included in the migration.
	
	For example:
	
	• To include only **User2** from the Fabrikam domain, enter:
	
	`/ue:*\* /ui:fabrikam\user2`
	
	• To migrate all users from the Fabrikam domain, and only the user accounts from other domains that are active or otherwise modified in the last 30 days, enter:
	
	`/uel:30 /ui:fabrikam\*`
	
	In this example, a user account from the Contoso domain that was last modified two months ago isn't migrated.
	
	For more examples, see the descriptions of the `/ue` and `/ui` options in this table.

/uel: *<NumberOfDays>* or **/uel**:*<YYYY/MM/DD>* or **/uel:0**	**(User exclude based on last logon)** Migrates the users that logged on to the source computer within the specified time period, based on the **Last Modified** date of the Ntuser.dat file on the source computer. The `/uel` option acts as an include rule. For example, the `/uel:30` option migrates users who logged on, or whose account was modified, within the last 30 days from the date when the `ScanState.exe` command is run. The number of days or the date can be specified. This option can't be used with the `/all` option. USMT retrieves the last sign-in information from the local computer, so the computer doesn't need to be connected to the network when running this option. In addition, if a domain user signs in to another computer, USMT doesn't consider that sign-in instance.

Command-Line Option	Description
	Note The `/uel` option isn't valid in offline migrations. • `/uel:0` migrates any users who are currently logged on. • `/uel:90` migrates users who logged on, or whose accounts were otherwise modified, within the last 90 days. • `/uel:1` migrates users whose account were modified within the last 24 hours. • `/uel:2020/2/15` migrates users who logged on or been modified February 15, 2020 or afterwards. For example: `ScanState.exe /i:MigApp.xml /i:MigDocs.xml \\server\share\migration\mystore /uel:0`
/ue:*<DomainName>**<UserName>* -or- **/ue**:*<ComputerName>**<LocalUserName>*	**(User exclude)** Excludes the specified users from the migration. Multiple `/ue` options can be specified. This option can't be used with the `/all` option. *<DomainName>* and *<UserName>* can contain the asterisk (`*`) wildcard character. When a user name that contains spaces is specified, it needs to be surrounded with quotation marks (`"`). For example: `ScanState.exe /i:MigDocs.xml /i:MigApp.xml \\server\share\migration\mystore /ue:contoso\user1`

How to use /ui and /ue

The following examples apply to both the `/ui` and `/ue` options. The `/ue` option can be replaced with the `/ui` option to include, rather than exclude, the specified users.

Expand table

Behavior	Command
Exclude the user named User One in the Fabrikam domain.	`/ue:"fabrikam\user one"`
Exclude the user named User1 in the Fabrikam domain.	`/ue:fabrikam\user1`
Exclude the local user named User1.	`/ue:%computername%\user1`

Behavior	Command
Exclude all domain users.	`/ue:Domain\*`
Exclude all local users.	`/ue:%computername%\*`
Exclude users in all domains named User1, User2, and so on.	`/ue:*\user*`

Using the options together

The `/uel`, `/ue` and `/ui` options can be used together to migrate only the users that need to be migrated.

The `/ui` option has precedence over the `/ue` and `/uel` options. If a user is specified for inclusion with the `/ui` option and also specified to be excluded with either the `/ue` or `/uel` options, the user is included in the migration. For example, if `/ui:contoso\*` `/ue:contoso\user1` is specified, then **User1** is migrated, because the `/ui` option takes precedence over the `/ue` option.

The `/uel` option takes precedence over the `/ue` option. If a user logged on within the specified time period set by the `/uel` option, that user's profile is migrated even if they're excluded by using the `/ue` option. For example, if `/ue:fixed\user1 /uel:14` is specified, then User1 is migrated if they logged on to the computer within the last 14 days.

Expand table

Behavior	Command
Include only User2 from the Fabrikam domain and exclude all other users.	`/ue:*\* /ui:fabrikam\user2`
Include only the local user named User1 and exclude all other users.	`/ue:*\* /ui:user1`
Include only the domain users from Contoso, except Contoso\User1.	This behavior can't be completed using a single command. Instead, to migrate this set of users, specify the following commands: • On the `ScanState.exe` command line, enter: `/ue:*\* /ui:contoso\*` • On the `LoadState.exe` command line, enter: `/ue:contoso\user1`

Behavior	Command
Include only local (non-domain) users.	`/ue:*\* /ui:%computername%\*`

Encrypted file options

The following options can be used to migrate encrypted files. In all cases, by default, USMT fails if an encrypted file is found unless the `/efs` option is specified. To migrate encrypted files, the default behavior must be changed.

For more information, see Migrate EFS Files and Certificates.

> **Note**
>
> EFS certificates are migrated automatically during the migration. Therefore, the `/efs:copyraw` option should be specified with the `ScanState.exe` command to migrate the encrypted files.

> **Caution**
>
> Take caution when migrating encrypted files. If an encrypted file is migrated without also migrating the certificate, end users won't be able to access the file after the migration.

Expand table

Command-Line Option	Explanation
/efs:hardlink	Creates a hard link to the EFS file instead of copying it. Use only with the `/hardlink` and the `/nocompress` options.
/efs:abort	Causes the `ScanState.exe` command to fail with an error code, if an Encrypting File System (EFS) file is found on the source computer. Enabled by default.
/efs:skip	Causes the `ScanState.exe` command to ignore EFS files.
/efs:decryptcopy	Causes the `ScanState.exe` command to decrypt the file, if possible, before saving it to the migration store, and to fail if the file can't be decrypted. If the `ScanState.exe` command succeeds, the file is unencrypted in the migration

Command-Line Option	Explanation
	store, and once the `LoadState.exe` command is run, the file is copied to the destination computer.
/efs:copyraw	Causes the `ScanState.exe` command to copy the files in the encrypted format. The files are inaccessible on the destination computer until the EFS certificates are migrated. EFS certificates are automatically migrated; however, by default USMT fails if an encrypted file is found, unless the `/efs` option is specified. Therefore the `/efs:copyraw` option should be specified with the `ScanState.exe` command to migrate the encrypted file. When the `LoadState.exe` command is run, the encrypted file and the EFS certificate are automatically migrated. For example: `ScanState.exe /i:MigDocs.xml /i:MigApp.xml \server\share\migration\mystore /efs:copyraw` **Important** All files must be encrypted if the parent folder is encrypted. If the encryption attribute on a file inside an encrypted folder is removed, the file is encrypted during the migration using the credentials of the account used to run the **LoadState** tool. For more information, see Migrate EFS files and certificates.

Incompatible command-line options

The following table indicates which command-line options aren't compatible with the `ScanState.exe` command. If the table entry for a particular combination has a ✅, the options are compatible and they can be used together. The ❌ symbol means that the options aren't compatible. For example, the `/nocompress` option can't be used with the `/encrypt` option.

Expand table

Command-Line Option	/keyfile	/nocompress	/genconfig	/all
/i	✅	✅	✅	✅
/o	✅	✅	✅	✅
/v	✅	✅	✅	✅

Command-Line Option	/keyfile	/nocompress	/genconfig	/all
/nocompress	✅	✅	✅	N/A
/localonly	✅	✅	❌	✅
/key	❌	✅	❌	✅
/encrypt	Required*	❌	❌	✅
/keyfile	N/A	✅	❌	✅
/l	✅	✅	✅	✅
/listfiles	✅	✅	❌	✅
/progress	✅	✅	❌	✅
/r	✅	✅	❌	✅
/w	✅	✅	❌	✅
/c	✅	✅	❌	✅
/p	✅	✅	❌	N/A
/all	✅	✅	❌	✅
/ui	✅	✅	❌	❌
/ue	✅	✅	❌	❌
/uel	✅	✅	❌	❌
/efs:*<option>*	✅	✅	❌	✅
/genconfig	✅	✅	N/A	✅
/config	✅	✅	❌	✅
<StorePath>	✅	✅	❌	✅

Note

Either the `/key` or `/keyfile` option must be specified with the `/encrypt` option.

Related articles

- XML Elements Library.

Feedback

Was this page helpful? Yes No

Provide product feedback

LoadState syntax

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

The `LoadState.exe` command is used with the User State Migration Tool (USMT) to restore a store previously captured by the `ScanState.exe` command onto a destination computer. This article discusses the `LoadState.exe` command syntax and the options available with it.

Before beginning

Before running the `LoadState.exe` command, note the following items:

- To ensure that all operating system settings migrate, Microsoft recommends running `LoadState.exe` commands in administrator mode from an account with administrative credentials.
- For information about software requirements for running the `LoadState.exe` command, see USMT requirements.
- Sign out after running the `LoadState.exe` command. Some settings, such as example, fonts, wallpaper, and screensaver settings, won't take effect until the next time the user logs in.
- Unless otherwise specified, each option can only be used once when running a tool from the command line.
- **LoadState** doesn't require domain controller access to apply domain profiles. This functionality is available without any additional configuration. It isn't necessary for the source computer to have domain controller access when the user profile was gathered using **ScanState**. However, domain profiles are inaccessible until the destination computer is joined to the domain.
- The Incompatible command-line options table lists which options can be used together and which command-line options are incompatible.

Syntax

This section explains the syntax and usage of the command-line options available when using the `LoadState.exe` command. The options can be specified in any order. If the option contains a parameter, either a colon or space separator can be specified.

The `LoadState.exe` command's syntax is:

> LoadState.exe *StorePath* [/i:[*Path*\]*FileName*] [/v:*VerbosityLevel*] [/nocompress] [/decrypt /key:*KeyString*|/keyfile:[Path\]*FileName*] [/l:[*Path*\]*FileName*] [/progress:[*Path*\]*FileName*] [/r:*TimesToRetry*] [/w:*SecondsToWait*] [/c] [/all] [/ui:[*DomainName*|*ComputerName*\]*UserName*] [/ue:[[*DomainName*|*ComputerName*\]*UserName*] [/uel:*NumberOfDays*|*YYYY/MM/DD*|0] [/md:*OldDomain*:*NewDomain*] [/mu:*OldDomain**OldUserName*:[*NewDomain*\]*NewUserName*] [/lac:[*Password*]] [/lae] [/config:[*Path*\]*FileName*] [/?|help]

For example, to decrypt the store and migrate the files and settings to a computer, type the following command:

`LoadState.exe \\server\share\migration\mystore /i:MigApp.xml /i:MigDocs.xml /v:13 /decrypt /key:"mykey"`

Storage options

USMT provides the following options that can be used to specify how and where the migrated data is stored.

Expand table

Command-Line Option	Description
StorePath	Indicates the folder where the files and settings data are stored. *StorePath* must be specified when using the `LoadState.exe` command. More than one *StorePath* can't be specified.
/decrypt /key:*KeyString* or **/decrypt /key**:"*Key String*" or **/decrypt /keyfile**:[*Path*]*FileName*	Decrypts the store with the specified key. With this option, the encryption key needs to be specified in one of the following ways: • `/key`:*KeyString* specifies the encryption key. If there's a space in *KeyString*, the argument must be surrounded with quotation marks (`"`). • `/keyfile`:*FilePathAndName* specifies a text (`.txt`) file that contains the encryption key *KeyString* can't exceed 256 characters. The `/key` and `/keyfile` options can't be used on the same command line. The `/decrypt` and `/nocompress` options can't be used on the same command line.

Command-Line Option	Description
	Important Use caution when using the `/key` or `keyfile` options. For example, anyone who has access to scripts that run the `LoadState.exe` command with these options also have access to the encryption key. For example: `LoadState.exe /i:MigApp.xml /i:MigDocs.xml \server\share\migration\mystore /decrypt /key:mykey`
/decrypt:"*encryption strength*"	The `/decrypt` option accepts a command-line parameter to define the encryption strength specified for the migration store encryption. For more information about supported encryption algorithms, see Migration Store Encryption.
/hardlink	Enables user-state data to be restored from a hard-link migration store. The `/nocompress` parameter must be specified with `/hardlink` option.
/nocompress	Specifies that the store isn't compressed. This option should only be used in testing environments. Microsoft recommends using a compressed store during the actual migration. This option can't be used with the `/decrypt` option. For example: `LoadState.exe /i:MigApp.xml /i:MigDocs.xml \server\share\migration\mystore /nocompress`

Migration rule options

USMT provides the following options to specify what files to migrate.

Expand table

Command-Line Option	Description
/i: [*Path*]*FileName*	**(include)** Specifies an **.xml** file that contains rules that define what data to migrate. This option can be specified multiple times to include all of the **.xml** files (`MigApp.xml`, `MigSys.xml`, `MigDocs.xml` and any custom **.xml** files that are created). *Path* can be either a relative or full path. If the *Path* variable isn't specified, then *FileName* must be located in the current directory.

Command-Line Option	Description
	For more information about which files to specify, see the "XML files" section of the Frequently Asked Questions article.
/config: [*Path*]*FileName*	Specifies the `Config.xml` file that the `LoadState.exe` command should use. This option can't be specified more than once on the command line. *Path* can be either a relative or full path. If the *Path* variable isn't specified, then the *FileName* must be located in the current directory. This example migrates the files and settings based on the rules in the `Config.xml`, `MigDocs.xml`, and `MigApp.xml` files: `LoadState.exe \server\share\migration\mystore /config:Config.xml /i:MigDocs.xml /i:MigApp.xml /v:5 /l:LoadState.log`
/auto:"*path to script files*"	This option enables specifying the location of the default **.xml** files. If no path is specified, USMT uses the directory where the USMT binaries are located. The `/auto` option has the same effect as using the following options: `/i:MigDocs.xml` `/i:MigApp.xml /v:5`.

Monitoring options

USMT provides several command-line options that can be used to analyze problems that occur during migration.

Expand table

Command-Line Option	Description
/l:[*Path*]*FileName*	Specifies the location and name of the **LoadState** log. The log files can't be stored in *StorePath*. *Path* can be either a relative or full path. If the *Path* variable isn't specified, then the log is created in the current directory. The `/v` option can be specified to adjust the verbosity of the log. If running the `LoadState.exe` command from a shared network resource, the `l` option must be specified, or USMT fails with the error: ***USMT was unable to create the log file(s)*** To fix this issue, make sure to specify the `/l` option when running `LoadState.exe` from a shared network resource.
/v:`<VerbosityLevel>`	**(Verbosity)**

Command-Line Option	Description
	Enables verbose output in the **LoadState** log file. The default value is 0. The *VerbosityLevel* can be set to one of the following levels: • **0** - Only the default errors and warnings are enabled. • **1** - Enables verbose output. • **4** - Enables error and status output. • **5** - Enables verbose and status output. • **8** - Enables error output to a debugger. • **9** - Enables verbose output to a debugger. • **12** - Enables error and status output to a debugger. • **13** - Enables verbose, status, and debugger output. For example: `LoadState.exe \server\share\migration\mystore /v:5 /i:MigDocs.xml /i:MigApp.xml`
/progress: [*Path*]*FileName*	Creates the optional progress log. The log files can't be stored in *StorePath*. *Path* can be either a relative or full path. If the *Path* variable isn't specified, then *FileName* is created in the current directory. For example: `LoadState.exe /i:MigApp.xml /i:MigDocs.xml \server\share\migration\mystore /progress:Progress.log /l:loadlog.log`
/c	When this option is specified, the `LoadState.exe` command continues to run, even if non-fatal errors occur. Any files or settings that cause an error are logged in the progress log. For example, if there's a large file that doesn't fit on the computer, the `LoadState.exe` command logs an error and continue with the migration. Without the `/c` option, the `LoadState.exe` command exits on the first error. The <**ErrorControl**> section can be used in the `Config.xml` file to specify which file or registry read/write errors can be safely ignored and which might cause the migration to fail. This error control enables the `/c` command-line option to safely skip all input/output (I/O) errors in the environment. In addition, the `/genconfig` option now generates a sample <**ErrorControl**> section that is enabled by specifying error messages and desired behaviors in the `Config.xml` file.
/r:`<TimesToRetry>`	**(Retry)** Specifies the number of times to retry when an error occurs while migrating the user state from a server. The default is three times. This option is useful in environments where network connectivity isn't reliable. When the user state is being restored, the `/r` option doesn't recover

Command-Line Option	Description
	data that is lost due to a network-hardware failure, such as a faulty or disconnected network cable, or when a virtual private network (VPN) connection fails. The retry option is intended for large, busy networks where connectivity is satisfactory, but communication latency is a problem.
/w: `<SecondsBeforeRetry>`	**(Wait)** Specifies the time to wait, in seconds, before retrying a network file operation. The default is 1 second.
/? or **/help**	Displays Help on the command line.

User options

By default, all users are migrated. The only way to specify which users to include and exclude is by using the following options. Users can't be excluded in the migration **.xml** files or by using the `Config.xml` file. For more information, see Identify Users.

Expand table

Command-Line Option	Description
/all	Migrates all of the users on the computer. USMT migrates all user accounts on the computer, unless an account is specifically excluded with the `/ue` or `/uel` options. For this reason, this option doesn't need to be specified on the command line. However, if using the `/all` option, the `/ui`, `/ue` or `/uel` options can't also be used.
/ui:*DomainName UserName* or **/ui**:"*DomainName User Name*" or **/ui**:*ComputerName LocalUserName*	**(User include)** Migrates the specified user. By default, all users are included in the migration. Therefore, this option is helpful only when used with the `/ue` option. Multiple `/ui` options can be specified, but the `/ui` option can't be used with the `/all` option. *DomainName* and *UserName* can contain the asterisk (`*`) wildcard character. When user name that contains spaces is specified, it needs to be surrounded with quotations marks (`"`).

Command-Line Option	Description
	For example, to include only **User2** from the Corporate domain, enter: `/ue:* /ui:corporate\user2` **Note** If a user is specified for inclusion with the `/ui` option and also specified to be excluded with either the `/ue` or `/uel` options, the user is included in the migration. For more examples, see the descriptions of the `/uel`, `/ue`, and `/ui` options in this table.
/uel:`<NumberOfDays>` or **/uel**:`<YYYY/MM/DD>` or **/uel**:0	**(User exclude based on last logon)** Migrates only the users that logged onto the source computer within the specified time period, based on the **Last Modified** date of the **Ntuser.dat** file on the source computer. The `/uel` option acts as an include rule. For example, the `/uel:30` option migrates users who logged on, or whose user account was modified, within the last 30 days from the date when the `ScanState.exe` command is run. The number of days can be specified or a date can be specified. This option can't be used with the `/all` option. USMT retrieves the last sign-in information from the local computer, so the computer doesn't need to be connected to the network when running this option. In addition, if a domain user signs into another computer, USMT doesn't consider that sign-in instance. **Note** The `/uel` option isn't valid in offline migrations. Examples: • `/uel:0` migrates accounts that were logged on to the source computer when the `ScanState.exe` command was run.

Command-Line Option	Description
	• `/uel:90` migrates users who logged on, or whose accounts were otherwise modified, within the last 90 days. • `/uel:1` migrates users whose accounts were modified within the last 24 hours. • `/uel:2020/2/15` migrates users who logged on or whose accounts modified since February 15, 2020. For example: `LoadState.exe /i:MigApp.xml /i:MigDocs.xml \server\share\migration\mystore /uel:0`
/ue:*DomainName\UserName* or **/ue** "*DomainName\User Name*" or **/ue**:*ComputerName\LocalUserName*	**(User exclude)** Excludes the specified users from the migration. Multiple `/ue` options can be used but the `/ue` option can't be used with the `/all` option. *DomainName* and *UserName* can contain the asterisk (`*`) wildcard character. When a user name that contains spaces is specified, it needs to be surround with quotation marks (`"`). For example: `LoadState.exe /i:MigApp.xml /i:MigDocs.xml \server\share\migration\mystore /ue:contoso\user1` For more examples, see the descriptions of the `/uel`, `/ue`, and `/ui` options in this table.
/md:*OldDomain*:*NewDomain* or **/md**:*LocalComputerName:NewDomain*	**(Move domain)** Specifies a new domain for the user. Use this option to change the domain for users on a computer or to migrate a local user to a domain account. *OldDomain* might contain the asterisk () wildcard character. This option can be specified more than once. If consolidating users across multiple domains to a single domain, multiple `/md` options might need to be specified. For example, to consolidate the users from the Corporate and FarNorth domains into the Fabrikam domain, specify the following settings: `/md:corporate:fabrikam` and `/md:farnorth:fabrikam`. If there are conflicts between two `/md` commands, the first rule specified is applied. For example, if the `/md:corporate:fabrikam` and `/md:corporate:farnorth`

Command-Line Option	Description
	commands are specified, then Corporate users would be mapped to the Fabrikam domain. **Note** If a domain that didn't exist on the source computer is specified, the `LoadState.exe` command appears to complete successfully, without an error or warning. However, in this case, users aren't moved to *NewDomain* but instead remain in their original domain. For example, if **contoso** is misspelled and instead **/md:contso:fabrikam** is specified, the users remain in **contoso** on the destination computer. For example: `LoadState.exe /i:MigApp.xml /i:MigDocs.xml \server\share\migration\mystore /progress:Progress.log /l:LoadState.log /md:contoso:fabrikam`
/mu:*OldDomain OldUserName*:[*NewDomain*]*NewUserName* or **/mu**:*OldLocalUserName*:*NewDomain NewUserName*	**(Move user)** Specifies a new user name for the specified user. If the store contains more than one user, multiple `/mu` options can be specified. Wildcard characters can't be used with this option. For example: `LoadState.exe /i:MigApp.xml /i:MigDocs.xml \server\share\migration\mystore /progress:Progress.log /l:LoadState.log /mu:contoso\user1:fabrikam\user1`
/lac:[*Password*]	**(Local account create)** If a user account is: • A local (non-domain) account • An account that doesn't exist on the destination computer this setting specifies to create the account on the destination computer. However, the account is disabled. To enable the account, the `/lae` option must also be

Command-Line Option	Description
	used. If the `/lac` option isn't specified, any local user accounts that don't already exist on the destination computer aren't migrated. *Password* is the password for the newly created account. An empty password is used by default. **Caution** Use the *Password* variable with caution. The *Password* variable is provided in plain text and anyone with access to the computer that is running the `LoadState.exe` command can obtain the password. Also, if the computer has multiple users, all migrated users have the same password. For example: `LoadState.exe /i:MigApp.xml /i:MigDocs.xml \server\share\migration\mystore` For instructions, see Migrate user accounts.
`/lae`	**(Local account enable)** Enables the account that was created with the `/lac` option. The `/lac` option must be specified with this option. For example: `LoadState.exe /i:MigApp.xml /i:MigDocs.xml \server\share\migration\mystore /progress:Progress.log /l:LoadState.log /lac:password /lae` For instructions, see Migrate user accounts.

Examples for the /ui and /ue options

The following examples apply to both the **/ui** and **/ue** options. The **/ue** option can be replaced with the **/ui** option to include, rather than exclude, the specified users.

Expand table

Behavior	Command
Exclude the user named User One in the Corporate domain.	`/ue:"corporate\user one"`
Exclude the user named User1 in the Corporate domain.	`/ue:corporate\user1`
Exclude the local user named User1.	`/ue:%computername%\user1`
Exclude all domain users.	`/ue:Domain`
Exclude all local users.	`/ue:%computername%`
Exclude users in all domains named User1, User2, and so on.	`/ue:\user`

Using the options together

The `/uel`, `/ue` and `/ui` options can be used together to migrate only the users that need to be migrated.

The /ui option has precedence over the /ue and /uel options. If a user is included using the `/ui` option and also excluded using either the `/ue` or `/uel` options, the user is included in the migration. For example, if `/ui:contoso\* /ue:contoso\user1` is specified, then User1 is migrated, because the `/ui` option takes precedence over the `/ue` option.

The /uel option takes precedence over the /ue option. If a user logged on within the specified time period set by the `/uel` option, that user's profile is migrated even if they're excluded by using the `/ue` option. For example, if `/ue:contoso\user1 /uel:14` is specified, then User1 is migrated if they logged on to the computer within the last 14 days.

Expand table

Behavior	Command
Include only User2 from the Fabrikam domain and exclude all other users.	`/ue:* /ui:fabrikam\user2`
Include only the local user named User1 and exclude all other users.	`/ue:* /ui:user1`
Include only the domain users from Contoso, except Contoso\User1.	This behavior can't be completed using a single command. Instead, to migrate this set of users, specify the following options:

Behavior	Command
	• Using the **ScanState** command-line tool, enter: `/ue:* /ui:contoso` • Using the **LoadState** command-line tool, enter: `/ue:contoso\user1`
Include only local (non-domain) users.	`/ue: /ui:%computername%*`

Incompatible command-line options

The following table indicates which command-line options aren't compatible with the `LoadState.exe` command. If the table entry for a particular combination has a ✅, the options are compatible, and they can be used together. The ❌ symbol means that the options aren't compatible. For example, the `/nocompress` option can't be used with the `/encrypt` option.

Expand table

Command-Line Option	/keyfile	/nocompress	/genconfig	/all
/i	✅	✅	✅	✅
/v	✅	✅	✅	✅
/nocompress	✅	N/A	❌	✅
/key	❌	✅	❌	✅
/decrypt	Required*	❌	❌	✅
/keyfile	N/A	✅	❌	✅
/l	✅	✅	✅	✅
/progress	✅	✅	❌	✅
/r	✅	✅	❌	✅
/w	✅	✅	❌	✅
/c	✅	✅	❌	✅
/p	✅	✅	❌	N/A
/all	✅	✅	❌	✅

Command-Line Option	/keyfile	/nocompress	/genconfig	/all
/ui	✅	✅	❌	❌
/ue	✅	✅	❌	❌
/uel	✅	✅	❌	❌
/genconfig	✅	✅	N/A	✅
/config	✅	✅	❌	✅
StorePath	✅	✅	✅	✅
/md	✅	✅	✅	✅
/mu	✅	✅	✅	✅
/lae	✅	✅	✅	✅
/lac	✅	✅	✅	✅

Note

Either the `/key` or `/keyfile` option must be specified with the `/decrypt` option.

Related articles

- XML elements library.

Feedback

Was this page helpful? Yes No

Provide product feedback

UsmtUtils Syntax

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

This article describes the syntax for the utilities available in User State Migration Tool (USMT) through the command-line interface. These utilities:

- Improve the ability to determine cryptographic options for the migration.
- Help removing hard-link stores that can't otherwise be deleted due to a sharing lock.
- Verify whether the catalog file or any of the other files in the compressed migration store are corrupted.
- Extract files from the compressed migration store created when files and settings are migrated to the destination computer.

UsmtUtils.exe

The following table lists command-line options for `UsmtUtils.exe`. The sections that follow provide further command-line options for the `/verify` and the `/extract` options.

The syntax for `UsmtUtils.exe` is:

> UsmtUtils.exe [/ec | /rd <*storeDir*> | /verify <*filepath*> [options] | /extract <*filepath*> <*destinationPath*> [options]]

Expand table

Command-line Option	Description
/ec	Returns a list of supported cryptographic algorithms (AlgIDs) on the current system. This option can be used on a destination computer to determine which algorithm to use with the `/encrypt` command before running the **ScanState** tool on the source computer.
/rd <*storeDir*>	Removes the directory path specified by the <*storeDir*> argument on the computer. This command can be used to delete hard-link migration stores that can't otherwise be deleted at a command prompt due to a sharing lock. If the migration store spans multiple volumes on a given drive, the migration store is deleted from all of these volumes.

Command-line Option	Description
	For example: `UsmtUtils.exe /rd D:\MyHardLinkStore`
/y	Overrides the prompt to accept deletions when used with the `/rd` option. When the `/y` option is used with the `/rd` option, a prompt isn't displayed to accept the deletions before USMT deletes the directories.
/verify	Returns information on whether the compressed migration store is intact or whether it contains corrupted files or a corrupted catalog. See Verify options for syntax and options to use with `/verify`.
/extract	Recovers files from a compressed USMT migration store. See Extract options for syntax and options to use with `/extract`.

Verify options

Use the `/verify` option to determine whether a compressed migration store is intact or whether it contains corrupted files or a corrupted catalog. For more information on how to use the `/verify` option, see Verify the condition of a compressed migration store.

The syntax for `/verify` is:

> UsmtUtils.exe /verify[:<*reportType*>] <*filePath*> [/l:<*logfile*>] [/v:*VerbosityLevel*] [/decrypt [:<*AlgID*>] {/key:<*keystring*> | /keyfile:<*filename*>}]

Expand table

Command-line Option	Description
<*reportType*>	Specifies whether to report on all files, corrupted files only, or the status of the catalog. • **Summary**. Returns both the number of files that are intact and the number of files that are corrupted in the migration store. If no algorithm is specified, the summary report is displayed as a default. • **all**. Returns a tab-delimited list of all of the files in the compressed migration store and the status for each file. Each line contains the file name followed by a tab spacing, and either **CORRUPTED** or **OK** depending on the status of the file. The last entry reports the corruption status of the **CATALOG** of the store. A catalog file contains metadata for all files in a migration store. The **LoadState** tool requires a valid catalog file in order to open the migration store. Returns "OK"

Command-line Option	Description
	if the catalog file is intact and **LoadState** can open the migration store and "CORRUPTED" if the migration store is corrupted. • **failureonly**. Returns a tab-delimited list of only the files that are corrupted in the compressed migration store. • **Catalog**. Returns only the status of the catalog file.
/l: *<logfilePath>*	Specifies the location and name of the log file.
/v: *<VerbosityLevel>*	**(Verbosity)** Enables verbose output in the **UsmtUtils** log file. The default value is 0. The *VerbosityLevel* can be set to one of the following levels: • **0** - Only the default errors and warnings are enabled. • **1** - Enables verbose output. • **4** - Enables error and status output. • **5** - Enables verbose and status output. • **8** - Enables error output to a debugger. • **9** - Enables verbose output to a debugger. • **12** - Enables error and status output to a debugger. • **13** - Enables verbose, status, and debugger output.
/decrypt *<AlgID>* **/:***<KeyString>* or **/decrypt** *<AlgID>* **/:***<"Key String">* or **/decrypt:** *<AlgID>* **/keyfile:** *<FileName>*	Specifies that the `/encrypt` option was used to create the migration store with the **ScanState** tool. To decrypt the migration store, specify a `/key` or `/keyfile` option as follows: • *<AlgID>* specifies the cryptographic algorithm that was used to create the migration store on the `ScanState.exe` command line. If no algorithm is specified, **ScanState** and **UsmtUtils** use the 3DES algorithm as a default. *<AlgID>* valid values include: `AES_128`, `AES_192`, `AES_256`, `3DES`, or `3DES_112`. • `/key:` *<KeyString>* specifies the encryption key. If there's a space in *<KeyString>*, the argument must be surrounded with quotation marks. • `/keyfile`: *<FileName>* specifies the location and name of a text (.txt) file that contains the encryption key. For more information about supported encryption algorithms, see Migration Store Encryption.

Some examples of `/verify` commands:

- `UsmtUtils.exe /verify D:\MyMigrationStore\store.mig`
- `UsmtUtils.exe /verify:catalog D:\MyMigrationStore\store.mig`
- `UsmtUtils.exe /verify:all D:\MyMigrationStore\store.mig /decrypt /l:D:\UsmtUtilsLog.txt`
- `UsmtUtils.exe /verify:failureonly D:\MyMigrationStore\store.mig /decrypt:AES_192 /keyfile:D:\encryptionKey.txt`

Extract options

Use the `/extract` option to recover files from a compressed USMT migration store if it doesn't restore normally with **LoadState**. For more information on how to use the `/extract` option, see Extract files from a compressed USMT migration store.

The syntax for `/extract` is:

> /extract <*filePath*> <*destinationPath*> [/i:<*includePattern*>] [/e: <*excludePattern*>] [/l: <*logfile*>] [/v: *VerbosityLevel*>] [/decrypt[:<*AlgID*>] {key: <*keystring*> | /keyfile: <*filename*>}] [/o]

Expand table

Command-line Option	Description
<*filePath*>	Path to the USMT migration store. For example: `D:\MyMigrationStore\USMT\store.mig`
<*destinationPath*>	Path to the folder where the tool puts the individual files.
/i:<*includePattern*>	Specifies a pattern for files to include in the extraction. More than one pattern can be specified. Separate patterns with a comma or a semicolon. The `/i`: <*includePattern*> and `/e`: <*excludePattern*> options can be used in the same command. When both include and exclude patterns are used on the command line, include patterns take precedence over exclude patterns.
/e:<*excludePattern*>	Specifies a pattern for files to omit from the extraction. More than one pattern can be specified. Separate patterns with a comma or a semicolon. The `/i`: <*includePattern*> and `/e`: <*excludePattern*> options can be used in the same command. When both include and exclude patterns are used

Command-line Option	Description
	on the command line, include patterns take precedence over exclude patterns.
/l:<*logfilePath*>	Specifies the location and name of the log file.
/v:<*VerbosityLevel*>	**(Verbosity)** Enables verbose output in the **UsmtUtils** log file. The default value is 0. The *VerbosityLevel* can be set to one of the following levels: • **0** - Only the default errors and warnings are enabled. • **1** - Enables verbose output. • **4** - Enables error and status output. • **5** - Enables verbose and status output. • **8** - Enables error output to a debugger. • **9** - Enables verbose output to a debugger. • **12** - Enables error and status output to a debugger. • **13** - Enables verbose, status, and debugger output.
/decrypt* <AlgID>***/key***: <KeyString> *or* ***/decrypt***<AlgID>***/***: <"Key String"> *or* ***/decrypt:*** <AlgID>***/keyfile***: <FileName>*	Specifies that the `/encrypt` option was used to create the migration store with the **ScanState** tool. To decrypt the migration store, the `/key` or `/keyfile` option must also be specified as follows: • <*AlgID*> specifies the cryptographic algorithm that was used to create the migration store on the `ScanState.exe` command line. If no algorithm is specified, **ScanState** and **UsmtUtils** use the 3DES algorithm as a default. <*AlgID*> valid values include: `AES_128`, `AES_192`, `AES_256`, `3DES`, or `3DES_112`. • `/key`: <*KeyString*> specifies the encryption key. If there's a space in <*KeyString*>, the argument must be surrounded with quotation marks. • `/keyfile`:<*FileName*> specifies a text (.txt) file that contains the encryption key For more information about supported encryption algorithms, see Migration store encryption.
/o	Overwrites existing output files.

Some examples of `/extract` commands:

- `UsmtUtils.exe /extract D:\MyMigrationStore\USMT\store.mig C:\ExtractedStore`

- `UsmtUtils.exe /extract D:\MyMigrationStore\USMT\store.mig /i:"*.txt, *.pdf" C:\ExtractedStore /decrypt /keyfile:D:\encryptionKey.txt`
- `UsmtUtils.exe /extract D:\MyMigrationStore\USMT\store.mig /e:*.exe C:\ExtractedStore /decrypt:AES_128 /key:password /l:C:\usmtlog.txt`
- `UsmtUtils.exe /extract D:\MyMigrationStore\USMT\store.mig /i:myProject.* /e:*.exe C:\ExtractedStore /o`

Related articles

- User State Migration Tool (USMT) command-line syntax.
- Return codes.

Feedback

Was this page helpful? Yes No

Provide product feedback

USMT XML reference

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

This section contains articles that can be used to work with and to customize the migration XML files.

In this section

Expand table

Link	Description
Understanding migration XML files	Provides an overview of the default and custom migration XML files and includes guidelines for creating and editing a customized version of the `MigDocs.xml` file.
Config.xml file	Describes the `Config.xml` file and policies concerning its configuration.
Customize USMT XML files	Describes how to customize USMT XML files.
Custom XML examples	Gives examples of XML files for various migration scenarios.
Conflicts and precedence	Describes the precedence of migration rules and how conflicts are handled.
General conventions	Describes the XML helper functions.
XML file requirements	Describes the requirements for custom XML files.
Recognized environment variables	Describes environment variables recognized by USMT.
XML elements library	Describes the XML elements and helper functions for authoring migration XML files to use with USMT.

Feedback

Was this page helpful? Yes No

Provide product feedback

Understanding migration XML files

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

The behavior of a basic User State Migration Tool (USMT) migration can be modified by using XML files. These files provide instructions on where and how the USMT tools should gather and apply files and settings. USMT includes three XML files that can be used to customize a basic migration: the `MigDocs.xml` and `MigUser.xml` files, which modify how files are discovered on the source computer, and the MigApps.xml file, which is required in order to migrate supported application settings. Custom XML files and a `Config.xml` file can be created and edited to further customize the migration.

This article provides an overview of the default and custom migration XML files and includes guidelines for creating and editing a customized version of the `MigDocs.xml` file. The `MigDocs.xml` file uses the new `GenerateDocPatterns` function available in USMT to automatically find user documents on a source computer.

Overview of the Config.xml file

The `Config.xml` file is the configuration file created by the `/genconfig` option of the **ScanState** tool. It can be used to modify which operating-system components USMT migrates. The `Config.xml` file can be used with other XML files, such as in the following example:

```
ScanState.exe /i:migapps.xml /i:MigDocs.xml /genconfig:c:\myFolder\Config.xml
```

When used this way, the `Config.xml` file tightly controls aspects of the migration, including user profiles, data, and settings, without modifying or creating other XML files. For more information about the `Config.xml` file, see Customize USMT XML Files and Config.xml File.

> ⓘ **Note**
>
> When modifying the XML elements in the `Config.xml` file, set the **migrate** property on an element to **no** instead of deleting the element from the file. If the element is deleted instead of setting the property, rules in other XML files can still migrate the component.

Overview of the MigApp.xml file

The `MigApp.xml` file installed with USMT includes instructions to migrate the settings for the applications listed in What Does USMT Migrate?. In order to migrate application settings, the `MigApp.xml` file must be included when using the **ScanState** and **LoadState** tools by using the `/i` option. The `MigDocs.xml` and `MigUser.xml` files don't migrate application settings. A custom XML file can be created to include additional applications. For more information, see Customize USMT XML Files.

> **Important**
>
> The `MigApps.xml` file only detects and migrates **.pst** files that are linked to Microsoft Office Outlook. For more information about migrating **.pst** files that aren't linked to Outlook, see Sample migration rules for customized versions of XML files.

Overview of the MigDocs.xml file

The `MigDocs.xml` file uses the new `GenerateDocPatterns` helper function to create instructions for USMT to migrate files from the source computer, based on the location of the files. The `MigDocs.xml` file can be used with the **ScanState** and **LoadState** tools to perform a more targeted migration than using USMT without XML instructions.

The default `MigDocs.xml` file migrates the following data:

- All files on the root of the drive except `%WINDIR%`, `%PROGRAMFILES%`, `%PROGRAMDATA%`, or `%USERS%`.
- All folders in the root directory of all fixed drives. For example: `c:\data_mail\*[*]`.
- All files from the root of the Profiles folder, except for files in the system profile. For example: `c:\users\name[mail.pst]`.
- All folders from the root of the Profiles folder, except for the system-profile folders. For example: `c:\users\name\new folder\*[*]`.
- Standard shared folders:
  - CSIDL_COMMON_DESKTOPDIRECTORY
  - CSIDL_COMMON_FAVORITES
  - CSIDL_COMMON_DOCUMENTS
  - CSIDL_COMMON_MUSIC

    - CSIDL_COMMON_PICTURES
    - CSIDL_COMMON_VIDEO
    - FOLDERID_PublicDownloads
- Standard user-profile folders for each user:
    - CSIDL_MYDOCUMENTS
    - CSIDL_MYPICTURES
    - FOLDERID_OriginalImages
    - CSIDL_MYMUSIC
    - CSIDL_MYVIDEO
    - CSIDL_FAVORITES
    - CSIDL_DESKTOP
    - CSIDL_QUICKLAUNCH
    - FOLDERID_Contacts
    - FOLDERID_Libraries
    - FOLDERID_Downloads
    - FOLDERID_SavedGames
    - FOLDERID_RecordedTV

The default `MigDocs.xml` file doesn't migrate the following data:

- Files tagged with both the **hidden** and **system** attributes.
- Files and folders on removable drives.
- Data from the %WINDIR%, %PROGRAMDATA%, and %PROGRAMFILES% folders.
- Folders that contain installed applications.

The `/genmigxml` option can be used with the **ScanState** tool to review and modify what files are migrated.

Overview of the MigUser.xml file

The `MigUser.xml` file includes instructions for USMT to migrate user files based on file name extensions. The `MigUser.xml` file can be used with the **ScanState** and **LoadState** tools to perform a more targeted migration than using USMT without XML instructions. The `MigUser.xml` file gathers all files from the standard user-profile folders, and any files on the computer with the specified file name extensions.

The default `MigUser.xml` file migrates the following data:

- All files from the standard user-profile folders, which are described as:
  - CSIDL_MYVIDEO
  - CSIDL_MYMUSIC
  - CSIDL_DESKTOP
  - CSIDL_STARTMENU
  - CSIDL_PERSONAL
  - CSIDL_MYPICTURES
  - CSIDL_FAVORITES
  - CSIDL_QUICK LAUNCH
- Files with the following extensions:

  `.accdb`, `.ch3`, `.csv`, `.dif`, `.doc*`, `.dot*`, `.dqy`, `.iqy`, `.mcw`, `.mdb*`, `.mpp`, `.one*`, `.oqy`, `.or6`, `.pot*`, `.ppa`, `.pps*`, `.ppt*`, `.pre`, `.pst`, `.pub`, `.qdf`, `.qel`, `.qph`, `.qsd`, `.rqy`, `.rtf`, `.scd`, `.sh3`, `.slk`, `.txt`, `.vl*`, `.vsd`, `.wk*`, `.wpd`, `.wps`, `.wq1`, `.wri`, `.xl*`, `.xla`, `.xlb`, `.xls*`

  > ⓘ **Note**
  >
  > The asterisk (`*`) stands for zero or more characters.

  > ⓘ **Note**
  >
  > The OpenDocument extensions (`*.odt`, `*.odp`, `*.ods`) that Microsoft Office applications can use aren't migrated by default.

The default `MigUser.xml` file doesn't migrate the following data:

- Files tagged with both the **Hidden** and **System** attributes.
- Files and folders on removable drives.
- Data from the `%WINDIR%`, `%PROGRAMFILES%`, `%PROGRAMDATA%` folders.
- ACLS for files in folders outside the user profile.

The `MigUser.xml` file can be copied and then the copy modified to include or exclude standard user-profile folders and file name extensions. If all of the extensions for the files that need to be migrated from the source computer are known, use the `MigUser.xml` file to move all of the relevant data, regardless of the location of the files. However, adding in all file extensions that need to be migrated to the `MigUser.xml` file can result in a migration that contains more files than intended. For example, if all **.jpg** files are migrated, it can also migrate image files such as thumbnails and logos from legacy applications that are installed on the source computer.

> **Note**
>
> Each file name extension included in the rules within the `MigUser.xml` file increases the amount of time needed for the **ScanState** tool to gather the files for the migration. If more than 300 file types are being migrated, the migration experience can be slow. For more information about other ways to organize the migration of the data, see the Using multiple XML files section of this article.

Using multiple XML files

Multiple XML files can be used with the **ScanState** and **LoadState** tools. Each of the default XML files included with or generated by USMT is configured for a specific component of the migration. Custom XML files can also be used to supplement these default files with more migration rules.

Expand table

XML migration file	Modifies the following components:
Config.xml file	Operating-system components such as desktop wallpaper and background theme. The `Config.xml` can also be extended to include some application and document settings by generating the `Config.xml` file with the other default

XML migration file	Modifies the following components:
	XML files. For more information, see Customize USMT XML Files and Config.xml File.
MigApps.xml file	Applications settings.
MigUser.xml or **MigDocs.xml** files	User files and profile settings.
Custom XML files	Application settings, user profile settings, or user files, beyond the rules contained in the other XML files.

For example, all of the XML migration file types can be used for a single migration, as in the following example:

Windows Command Prompt

```
ScanState.exe <store> /config:c:\myFolder\Config.xml /i:migapps.xml
/i:MigDocs.xml /i:CustomRules.xml
```

XML rules for migrating user files

> **Important**
>
> The `MigUser.xml` and `MigDocs.xml` files shouldn't be used together in the same command. Using both XML files can result in duplication of some migrated files. Duplication of some migrated files can occur when conflicting target-location instructions are given in each XML file. The target file is stored once during the migration, but each XML file applies the file to a different location on the destination computer.

If the data set is unknown or if many files are stored outside of the standard user-profile folders, the `MigDocs.xml` is a better choice than the `MigUser.xml` file, because the `MigDocs.xml` file gathers a broader scope of data. The `MigDocs.xml` file migrates folders of data based on location. The `MigUser.xml` file migrates only the files with the specified file name extensions.

For more control over the migration, create custom XML files. For more information on creating custom XML files, see Creating and editing a custom XML file.

Creating and editing a custom XML file

The `/genmigxml` command-line option can be used to determine which files are included in the migration. The `/genmigxml` option creates a file in a specified location. The XML rules in the file can then be reviewed and if necessary, modifications made.

> **Note**
>
> If USMT is reinstalled, the default migration XML files are overwritten and any customizations made to these files are lost. Consider creating separate XML files for the custom migration rules and saving them in a secure location.

To generate the XML migration rules file for a source computer:

1. Select **Start** > **All Programs** > **Accessories**
2. Right-click **Command Prompt**, and then select **Run as**.
3. Select an account with administrator privileges, supply a password, and then select **OK**.
4. At the command prompt, enter:

   Windows Command Prompt

   ```
   cd /d <USMTpath>
   ScanState.exe /genmigxml: <filepath.xml>
   ```

   where:

   - **<USMTpath>** - location on the source computer of the saved USMT files and tools.
   - **<filepath.xml>** - full path to a file where the report can be saved.

   For example, enter:

   Windows Command Prompt

   ```
   cd /d c:\USMT
   ScanState.exe /genmigxml:"C:\Users\USMT Tester\Desktop\genMig.xml"
   ```

The GenerateDocPatterns function

The `MigDocs.xml` file calls the `GenerateDocPatterns` function, which takes three Boolean values. The settings can be changed to modify the way the `MigDocs.xml` file generates the XML rules for migration.

- `ScanProgramFiles`: This argument is valid only when the `GenerateDocPatterns` function is called in a system context. This argument determines whether or not to scan the Program Files directory to gather registered file name extensions for known applications.

  Default value: False

  For example, when set to **TRUE**, the function discovers and migrates **.doc** files under the Microsoft Office directory, because **.doc** is a file name extension registered to a Microsoft Office application. The `GenerateDocPatterns` function generates this inclusion pattern for `.doc` files:

  `<pattern type="File">C:\Program Files\Microsoft Office[.doc]</pattern>`

  If a child folder of an included folder contains an installed application, `ScanProgramFiles` also creates an exclusion rule for the child folder. All folders under the application folder are scanned recursively for registered file name extensions.

- `IncludePatterns`: This argument determines whether to generate exclude or include patterns in the XML. When this argument is set to **TRUE**, the `GenerateDocPatterns` function generates include patterns, and the function must be added under the `<include>` element. Changing this argument to **FALSE** generates exclude patterns and the function must be added under the `<exclude>` element.

  Default value: True

- `SystemDrive`: This argument determines whether to generate patterns for all fixed drives or only for the system drive. Changing this argument to **TRUE** restricts all patterns to the system drive.

  Default value: False

Usage:

Windows Command Prompt

```
MigXmlHelper.GenerateDocPatterns ("<ScanProgramFiles>", "<IncludePatterns>",
"<SystemDrive>")
```

To create include data patterns for only the system drive:

```XML
<include filter='MigXmlHelper.IgnoreIrrelevantLinks()'>
     <objectSet>
        <script>MigXmlHelper.GenerateDocPatterns ("FALSE","TRUE","TRUE")
</script>
     </objectSet>
</include>
```

To create an include rule to gather files for registered extensions from the %PROGRAMFILES% directory:

```XML
<include filter='MigXmlHelper.IgnoreIrrelevantLinks()'>
     <objectSet>
        <script>MigXmlHelper.GenerateDocPatterns ("TRUE","TRUE","FALSE")
</script>
     </objectSet>
</include>
```

To create exclude data patterns:

```XML
<exclude filter='MigXmlHelper.IgnoreIrrelevantLinks()'>
     <objectSet>
        <script>MigXmlHelper.GenerateDocPatterns ("FALSE","FALSE","FALSE")
</script>
     </objectSet>
</exclude>
```

Understanding the system and user context

The migration XML files contain two <component> elements with different **context** settings:

- The system context applies to files on the computer that aren't stored in the User Profiles directory.
- The user context applies to files that are particular to an individual user.

System context

The system context includes rules for data outside of the User Profiles directory. For example, when called in a system context in the `MigDocs.xml` file, the `GenerateDocPatterns` function creates patterns for all common shell folders, files in the root directory of hard drives, and folders located at the root of hard drives. The following folders are included:

- CSIDL_COMMON_DESKTOPDIRECTORY
- CSIDL_COMMON_FAVORITES
- CSIDL_COMMON_DOCUMENTS
- CSIDL_COMMON_MUSIC
- CSIDL_COMMON_PICTURES
- CSIDL_COMMON_VIDEO
- FOLDERID_PublicDownloads

User context

The user context includes rules for data in the User Profiles directory. When called in a user context in the `MigDocs.xml` file, the `GenerateDocPatterns` function creates patterns for all user shell folders, files located at the root of the profile, and folders located at the root of the profile. The following folders are included:

- CSIDL_MYDOCUMENTS
- CSIDL_MYPICTURES
- FOLDERID_OriginalImages
- CSIDL_MYMUSIC
- CSIDL_MYVIDEO
- CSIDL_FAVORITES
- CSIDL_DESKTOP
- CSIDL_QUICKLAUNCH
- FOLDERID_Contacts
- FOLDERID_Libraries

- FOLDERID_Downloads
- FOLDERID_SavedGames
- FOLDERID_RecordedTV

> **Note**
>
> Rules contained in a component that is assigned the user context runs for each user profile on the computer. Files that are scanned multiple times by the `MigDocs.xml` files are only copied to the migration store once. However, a large number of rules in the user context can slow down the migration. Use the system context when it's applicable.

Sample migration rules for customized versions of XML files

> **Tip**
>
> For best practices and requirements for customized XML files in USMT, see Customize USMT XML Files and General Conventions.

Exclude rules usage examples

In the following examples, the source computer has a **.txt** file called `new text document` in a directory called `new folder`. The default `MigDocs.xml` behavior migrates the new text `document.txt` file and all files contained in the `new folder` directory. The rules generated by the function are:

Expand table

Rule	Syntax
Rule 1	`<pattern type="File">d:\new folder[new text document.txt]</pattern>`
Rule 2	`<pattern type="File">d:\new folder[]</pattern>`

To exclude the new text `document.txt` file and any **.txt** files in `new folder`, the following modifications can be made:

Example 1: Exclude all .txt files in a folder

To exclude Rule 1, there needs to be an exact match of the file name. However, for Rule 2, a pattern can be created to exclude files by using the file name extension.

XML

```xml
<exclude>
     <objectSet>
        <pattern type="File">D:\Newfolder\[new text document.txt]</pattern>
         <pattern type="File">D:\New folder\*[*.txt]</pattern>
     </objectSet>
</exclude>
```

Example 2: Use the UnconditionalExclude element to give a rule precedence over include rules

If the file name or location of the file isn't known, but the file name extension is known, the `GenerateDrivePatterns` function can be used. However, the rule is less specific than the default include rule generated by the `MigDocs.xml` file, so it doesn't have precedence. The <UnconditionalExclude> element must be used to give this rule precedence over the default include rule. For more information about the order of precedence for XML migration rules, see Conflicts and Precedence.

XML

```xml
<unconditionalExclude>
     <objectSet>
        <script>MigXmlHelper.GenerateDrivePatterns ("*[*.txt]", "Fixed")
</script>
     </objectSet>
</unconditionalExclude>
```

Example 3: Use a UserandSystem context component to run rules in both contexts

To apply the **<UnconditionalExclude>** element to both the system and user context, a third component can be created using the **UserandSystem** context. Rules in this component run in both contexts.

XML

```xml
<component type="Documents" context="UserandSystem">
   <displayName>MigDocExcludes</displayName>
```

```
    <role role="Data">
      <rules>
        <unconditionalExclude>
          <objectSet>
                 <script>MigXmlHelper.GenerateDrivePatterns ("*[*.txt]",
"Fixed")</script>
          </objectSet>
        </unconditionalExclude>
      </rules>
    </role>
</component>
```

For more examples of exclude rules that can be used in custom migration XML files, see Exclude Files and Settings.

Include rules usage examples

The application data directory is the most common location that an include rule would need to be added for. The `GenerateDocPatterns` function excludes this location by default. If the organization uses an application that saves important data to this location, include rules can be created to migrate the data. For example, the default location for **.pst** files is: `%CSIDL_LOCAL_APPDATA%\Microsoft\Outlook`. The `MigApp.xml` file contains migration rules to move only those **.pst** files that are linked to Microsoft Outlook. To include **.pst** files that aren't linked, the following modification can be made:

Example 1: Include a file name extension in a known user folder

This rule includes **.pst** files that are located in the default location, but aren't linked to Microsoft Outlook. Use the user context to run this rule for each user on the computer.

XML

```
<include filter='MigXmlHelper.IgnoreIrrelevantLinks()'>
     <objectSet>
        <pattern type="File">%CSIDL_LOCAL_APPDATA%\Microsoft\Outlook\*
[*.pst]</pattern>
     </objectSet>
</include>
```

Example 2: Include a file name extension in Program Files

For locations outside the user profile, such as the Program Files folder, the rule can be added to the system context component.

```XML
<include filter='MigXmlHelper.IgnoreIrrelevantLinks()'>
     <objectSet>
        <pattern type="File">%CSIDL_PROGRAM_FILES%\*[*.pst]</pattern>
     </objectSet>
</include>
```

For more examples of include rules that can be used in custom migration XML files, see Include Files and Settings.

> **Tip**
>
> For more information about the order of precedence for XML migration rules, see Conflicts and Precedence.

Next steps

Additional rules for the migration can be included in the `MigDocs.xml` file or other XML migration files. For example, the `<locationModify>` element can be used to move files from the folder where they were gathered to a different folder, when they're applied to the destination computer.

An XML schema (`MigXML.xsd`) file can be used to validate the syntax of the customized XML files. For more information, see USMT Resources.

Related articles

- Exclude files and settings.
- Include files and settings.

Feedback

Was this page helpful? Yes No

Provide product feedback

Config.xml File

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

The `Config.xml` file is an optional User State Migration Tool (USMT) file that can be created using the `/genconfig` option with the **ScanState** tool. If all of the default components should be included and no changes need to be made to the default store-creation or profile-migration behavior, a `Config.xml` file doesn't need to be created.

However, if the default migration behavior defined in the `MigApp.xml`, `MigUser.xml` and `MigDocs.xml` files is satisfactory, but certain components need to be excluded, a `Config.xml` file can be created and modified while leaving the other **.xml** files unchanged. For example, a `Config.xml` file must be created to exclude any of the operating-system settings that are migrated. It's necessary to create and modify the `Config.xml` file to change any of the default store-creation or profile-migration behavior.

The `Config.xml` file has a different format than the other migration **.xml** files, because it doesn't contain any migration rules. It contains only a list of the operating-system components, applications, user documents that can be migrated, and user-profile policy and error-control policy. For this reason, excluding components using the `Config.xml` file is easier than modifying the migration **.xml** files, because familiarization with the migration rules and syntax isn't needed. However, wildcard characters can't be used in this file.

For more information about using the `Config.xml` file with other migration files, such as the `MigDocs.xml` and `MigApps.xml` files, see Understanding Migration XML Files.

> ⓘ **Note**
>
> To exclude a component from the `Config.xml` file, set the **migrate** value to **no**. Deleting the XML tag for the component from the `Config.xml` file doesn't exclude the component from the migration.

Migration Policies

In USMT, there are migration policies that can be configured in the `Config.xml` file. For example, <**ErrorControl**>, <**ProfileControl**>, and <**HardLinkStoreControl**> options can be configured. The following elements and parameters are for use in the `Config.xml` file only.

<Policies>

The <**Policies**> element contains elements that describe the policies that USMT follows while creating a migration store. Valid children of the <**Policies**> element are <**ErrorControl**> and <**HardLinkStoreControl**>. The <**Policies**> element is a child of <**Configuration**>.

Syntax:

```XML
<Policies> </Policies>
```

<ErrorControl>

The <**ErrorControl**> element is an optional element that can be configured in the `Config.xml` file. The configurable <**ErrorControl**> rules support only the environment variables for the operating system that is running and the currently logged-on user. As a workaround, a path can be specified using the (*) wildcard character.

- **Number of occurrences**: Once for each component
- **Parent elements**: The <**Policies**> element
- **Child elements**: The <**fileError**> and <**registryError**> element

Syntax:

```XML
<ErrorControl> </ErrorControl>
```

The following example specifies that all locked files, regardless of their location (including files in C:\Users), should be ignored. However, the migration fails if any file in C:\Users can't be accessed because of any other reason. In the following example, the <**ErrorControl**> element ignores any problems in migrating registry keys that match the supplied pattern, and it resolves them to an **Access denied** error.

Additionally, the order in the <**ErrorControl**> section implies priority. In this example, the first <**nonFatal**> tag takes precedence over the second <**fatal**> tag. This precedence is applied, regardless of how many tags are listed.

```XML
```

```
<ErrorControl>
  <fileError>
    <nonFatal errorCode="33">* [*]</nonFatal>
    <fatal errorCode="any">C:\Users\* [*]</fatal>
  </fileError>
  <registryError>
    <nonFatal errorCode="5">HKCU\SOFTWARE\Microsoft\* [*]</nonFatal>
  </registryError>
</ErrorControl>
```

> **Important**
>
> The configurable **<ErrorControl>** rules support only the environment variables for the operating system that is running and the currently logged-on user. As a workaround, a path using the (*) wildcard character can be specified.

<fatal>

The **<fatal>** element isn't required.

- **Number of occurrences**: Once for each component
- **Parent elements**: **<fileError>** and **<registryError>**
- **Child elements**: None.

Syntax:

XML

```
<fatal errorCode="any"> <specify pattern here> </fatal>
```

Expand table

Parameter	Required	Value
errorCode	No	"any" or "*specify system error message here*"

The **<fatal>** element can be used to specify that errors matching a specific pattern should cause USMT to halt the migration.

<fileError>

The <**fileError**> element isn't required.

- **Number of occurrences**: Once for each component
- **Parent elements**: <**ErrorControl**>
- **Child elements**: <**nonFatal**> and <**fatal**>

Syntax:

```XML
<fileError> </fileError>
```

The <**fileError**> element can be used to represent the behavior associated with file errors.

<nonFatal>

The <**nonFatal**> element isn't required.

- **Number of occurrences**: Once for each component
- **Parent elements**: The <**fileError**> and <**registryError**> elements.
- **Child elements**: None.

Syntax:

```XML
<nonfatal errorCode="any"> <specify pattern here> </nonFatal>
```

Expand table

Parameter	Required	Value
<**errorCode**>	No	"any" or "*specify system error message*". If system error messages aren't specified, the default behavior applies the parameter to all system error messages.

The <**nonFatal**> element can be used to specify that errors matching a specific pattern shouldn't cause USMT to halt the migration.

<registryError>

The **<registryError>** element isn't required.

- **Number of occurrences**: Once for each component
- **Parent elements**: **<ErrorControl>**
- **Child elements**: **<nonfatal>** and **<fatal>**

Syntax:

```XML
<registryError errorcode="any"> </registryError>
```

Expand table

Parameter	Required	Value
<errorCode>	No	"any" or "*specify system error message here*". If system error messages aren't specified, the default behavior applies the parameter to all system error messages.

The **<registryError>** element can be used to specify that errors matching a specific pattern shouldn't cause USMT to halt the migration.

<HardLinkStoreControl>

The **<HardLinkStoreControl>** element contains elements that describe how to handle files during the creation of a hard-link migration store. Its only valid child is **<fileLocked>**.

Syntax:

```XML
<HardLinkStoreControl> </HardLinkStoreControl>
```

- **Number of occurrences**: Once for each component
- **Parent elements**: **<Policies>**
- **Child elements**: **<fileLocked>**

Syntax:

```XML
<HardLinkStoreControl> </HardLinkStoreControl>
```

The following **<HardLinkStoreControl>** sample code specifies that hard links can be created to locked files only if the locked file resides somewhere under C:\Users\. Otherwise, a file-access error occurs when a locked file is encountered that can't be copied, even though is technically possible for the link to be created.

> **Important**
>
> The **<ErrorControl>** section can be configured to conditionally ignore file access errors, based on the file's location.

```XML
<Policy>
   <HardLinkStoreControl>
      <fileLocked>
         <createHardLink>C:\Users\*</createHardLink>
         <errorHardLink>C:\*</errorHardLink>
      </fileLocked>
   </HardLinkStoreControl>
   <ErrorControl>
      […]
   </ErrorControl>
</Policy>
```

<fileLocked>

The **<fileLocked>** element contains elements that describe how to handle files that are locked for editing. The rules defined by the **<fileLocked>** element are processed in the order in which they appear in the XML file.

Syntax:

```XML
<fileLocked> </fileLocked>
```

<createHardLink>

The <**createHardLink**> element defines a standard MigXML pattern that describes file paths where hard links should be created, even if the file is locked for editing by another application.

Syntax:

```XML
<createHardLink> <specify pattern here> </createHardLink>
```

<errorHardLink>

The <**errorHardLink**> element defines a standard MigXML pattern that describes file paths where hard links shouldn't be created if the file is locked for editing by another application. USMT attempts to copy files under these paths into the migration store. However, if that isn't possible, **Error_Locked** is thrown. This error is a standard Windows application programming interface (API) error that can be captured by the <**ErrorControl**> section to either cause USMT to skip the file or abort the migration.

Syntax:

```XML
<errorHardLink> <specify pattern here> </errorHardLink>
```

<ProfileControl>

This element is used to contain other elements that establish rules for migrating profiles, users, and policies around local group membership during the migration. <**ProfileMigration**> is a child of <**Configuration**>.

Syntax:

```XML
<ProfileControl> </ProfileControl>
```

<localGroups>

This element is used to contain other elements that establish rules for how to migrate local groups. <**localGroups**> is a child of <**ProfileControl**>.

Syntax:

```XML
<localGroups> </localGroups>
```

<mappings>

This element is used to contain other elements that establish mappings between groups.

Syntax:

```XML
<mappings> </mappings>
```

<changeGroup>

This element describes the source and destination groups for a local group membership change during the migration. It's a child of **<localGroups>**. The following parameters are defined:

Expand table

Parameter	Required	Value
From	Yes	A valid local group on the source machine that contains users selected for migration on the command line.
To	Yes	A local group that the users are to be moved to during the migration.
appliesTo	Yes	nonmigratedUsers, migratedUsers, AllUsers. This value defines which users the change group operation should apply to.

The valid and required children of **<changeGroup>** are **<include>** and **<exclude>**. Although both can be children at the same time, only one is required.

Syntax:

```XML
<changeGroup From="Group1" To= "Group2"> </changeGroup>
```

<include>

This element specifies that its required child, <*pattern*>, should be included in the migration.

Syntax:

```XML
<include> </include>
```

<exclude>

This element specifies that its required child, <*pattern*>, should be excluded from the migration.

Syntax:

```XML
<exclude> </exclude>
```

Sample Config.xml File

The following sample `Config.xml` file contains detailed examples about items that can be excluded from a migration.

▼ Expand for sample Config.xml file:

```XML
<?xml version="1.0" encoding="UTF-8"?>
<Configuration>
  <Applications/>
  <Documents/>
  <WindowsComponents>
    <component displayname="Tablet PC Settings" migrate="yes"
ID="tablet_pc_settings">
      <component displayname="Accessories" migrate="yes"
ID="tablet_pc_settings\tablet_pc_accessories">
        <component displayname="Microsoft-Windows-TabletPC-StickyNotes"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
tabletpc-stickynotes/microsoft-windows-tabletpc-stickynotes/settings"/>
```

```
        <component displayname="Microsoft-Windows-TabletPC-SnippingTool"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
tabletpc-snippingtool/microsoft-windows-tabletpc-snippingtool/settings"/>
        <component displayname="Microsoft-Windows-TabletPC-Journal"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
tabletpc-journal/microsoft-windows-tabletpc-journal/settings"/>
      </component>
      <component displayname="Input Panel" migrate="yes"
ID="tablet_pc_settings\tablet_pc_input_panel">
        <component displayname="Microsoft-Windows-TabletPC-InputPanel"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
tabletpc-inputpanel/microsoft-windows-tabletpc-inputpanel/settings"/>
      </component>
      <component displayname="General Options" migrate="yes"
ID="tablet_pc_settings\tablet_pc_general_options">
        <component displayname="Microsoft-Windows-TabletPC-UIHub"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
tabletpc-uihub/microsoft-windows-tabletpc-uihub/settings"/>
        <component displayname="Microsoft-Windows-TabletPC-Platform-Input-
Core" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
tabletpc-platform-input-core/microsoft-windows-tabletpc-platform-input-
core/settings"/>
      </component>
      <component displayname="Handwriting Recognition" migrate="yes"
ID="tablet_pc_settings\handwriting_recognition">
        <component displayname="Microsoft-Windows-TabletPC-
InputPersonalization" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
tabletpc-inputpersonalization/microsoft-windows-tabletpc-
inputpersonalization/settings"/>
      </component>
    </component>
    <component displayname="Sound and Speech Recognition" migrate="yes"
ID="sound_and_speech_recognition">
      <component displayname="Speech Recognition" migrate="yes"
ID="sound_and_speech_recognition\speech_recognition">
        <component displayname="Microsoft-Windows-SpeechCommon"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
speechcommon/microsoft-windows-speechcommon/settings"/>
      </component>
    </component>
    <component displayname="Hardware" migrate="yes" ID="hardware">
      <component displayname="Phone and Modem" migrate="yes"
ID="hardware\phone_and_modem">
        <component displayname="Microsoft-Windows-TapiSetup" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
tapisetup/microsoft-windows-tapisetup/settings"/>
      </component>
      <component displayname="Printers and Faxes" migrate="yes"
```

```
ID="hardware\printers_and_faxes">
        <component displayname="Microsoft-Windows-Printing-Spooler-Core"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
printing-spooler-core/microsoft-windows-printing-spooler-core/settings"/>
        <component displayname="Microsoft-Windows-Printing-Spooler-
Networkclient" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
printing-spooler-networkclient/microsoft-windows-printing-spooler-
networkclient/settings"/>
        <component displayname="Microsoft-Windows-Printing-Spooler-Core-
Localspl" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
printing-spooler-core-localspl/microsoft-windows-printing-spooler-core-
localspl/settings"/>
      </component>
    </component>
    <component displayname="Programs" migrate="yes" ID="programs">
      <component displayname="Media Player Settings" migrate="yes"
ID="programs\media_player_settings">
        <component displayname="Microsoft-Windows-MediaPlayer-Migration"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
mediaplayer-migration/microsoft-windows-mediaplayer-migration/settings"/>
      </component>
    </component>
    <component displayname="Communications and Sync" migrate="yes"
ID="communications_and_sync">
      <component displayname="Windows Mail" migrate="yes"
ID="communications_and_sync\windows_mail">
        <component displayname="Microsoft-Windows-WAB" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
wab/microsoft-windows-wab/settings"/>
        <component displayname="Microsoft-Windows-Mail" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
mail/microsoft-windows-mail/settings"/>
      </component>
    </component>
    <component displayname="Microsoft-Windows-Migration-DisplayGroups"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
migration-displaygroups/microsoft-windows-migration-
displaygroups/settings"/>
    <component displayname="Performance and Maintenance" migrate="yes"
ID="performance_and_maintenance">
      <component displayname="Diagnostics" migrate="yes"
ID="performance_and_maintenance\diagnostics">
        <component displayname="Microsoft-Windows-RemoteAssistance-Exe"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
remoteassistance-exe/microsoft-windows-remoteassistance-exe/settings"/>
        <component displayname="Microsoft-Windows-Feedback-Service"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
feedback-service/microsoft-windows-feedback-service/settings"/>
```

```
      </component>
      <component displayname="Error Reporting" migrate="yes"
ID="performance_and_maintenance\error_reporting">
        <component displayname="Microsoft-Windows-ErrorReportingCore"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
errorreportingcore/microsoft-windows-errorreportingcore/settings"/>
      </component>
    </component>
    <component displayname="Network and Internet" migrate="yes"
ID="network_and_internet">
      <component displayname="Offline Files" migrate="yes"
ID="network_and_internet\offline_files">
        <component displayname="Microsoft-Windows-OfflineFiles-Core"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
offlinefiles-core/microsoft-windows-offlinefiles-core/settings"/>
      </component>
      <component displayname="Internet Options" migrate="yes"
ID="network_and_internet\internet_options">
        <component displayname="Microsoft-Windows-ieframe" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
ieframe/microsoft-windows-ieframe/settings"/>
        <component displayname="Microsoft-Windows-IE-InternetExplorer"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
ie-internetexplorer/microsoft-windows-ie-internetexplorer/settings"/>
        <component displayname="Microsoft-Windows-IE-Feeds-Platform"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
ie-feeds-platform/microsoft-windows-ie-feeds-platform/settings"/>
        <component displayname="Microsoft-Windows-IE-
ClientNetworkProtocolImplementation" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
ie-clientnetworkprotocolimplementation/microsoft-windows-ie-
clientnetworkprotocolimplementation/settings"/>
      </component>
      <component displayname="Networking Connections" migrate="yes"
ID="network_and_internet\networking_connections">
        <component displayname="Microsoft-Windows-Wlansvc" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
wlansvc/microsoft-windows-wlansvc/settings"/>
        <component displayname="Microsoft-Windows-RasConnectionManager"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
rasconnectionmanager/microsoft-windows-rasconnectionmanager/settings"/>
        <component displayname="Microsoft-Windows-RasApi" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
rasapi/microsoft-windows-rasapi/settings"/>
        <component displayname="Microsoft-Windows-PeerToPeerCollab"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
peertopeercollab/microsoft-windows-peertopeercollab/settings"/>
        <component displayname="Microsoft-Windows-MPR" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
```

```
mpr/microsoft-windows-mpr/settings"/>
        <component displayname="Microsoft-Windows-Dot3svc" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
dot3svc/microsoft-windows-dot3svc/settings"/>
      </component>
    </component>
    <component displayname="Date, Time, Language and Region" migrate="yes"
ID="date_time_language_and_region">
      <component displayname="Regional Language Options" migrate="yes"
ID="date_time_language_and_region\regional_language_options">
        <component displayname="Microsoft-Windows-TableDrivenTextService-
Migration" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
tabledriventextservice-migration/microsoft-windows-tabledriventextservice-
migration/settings"/>
        <component displayname="Microsoft-Windows-TextServicesFramework-
Migration" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
textservicesframework-migration/microsoft-windows-textservicesframework-
migration/settings"/>
        <component displayname="Microsoft-Windows-MUI-Settings"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
mui-settings/microsoft-windows-mui-settings/settings"/>
        <component displayname="Microsoft-Windows-International-Core"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
international-core/microsoft-windows-international-core/settings"/>
        <component displayname="Microsoft-Windows-IME-Traditional-Chinese-
Core" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
ime-traditional-chinese-core/microsoft-windows-ime-traditional-chinese-
core/settings"/>
        <component displayname="Microsoft-Windows-IME-Simplified-Chinese-
Core" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
ime-simplified-chinese-core/microsoft-windows-ime-simplified-chinese-
core/settings"/>
        <component displayname="Microsoft-Windows-Desktop_Technologies-
Text_Input_Services-IME-Japanese-Core" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
desktop_technologies-text_input_services-ime-japanese-core/microsoft-
windows-desktop_technologies-text_input_services-ime-japanese-
core/settings"/>
      </component>
    </component>
    <component displayname="Security" migrate="yes" ID="security">
      <component displayname="Microsoft-Windows-Rights-Management-Client-v1-
API" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
rights-management-client-v1-api/microsoft-windows-rights-management-client-
v1-api/settings"/>
      <component displayname="Security Options" migrate="yes"
ID="security\security_options">
        <component displayname="Microsoft-Windows-Credential-Manager"
```

```
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
credential-manager/microsoft-windows-credential-manager/settings"/>
      </component>
    </component>
    <component displayname="Appearance and Display" migrate="yes"
ID="appearance_and_display">
      <component displayname="Windows Games Settings" migrate="yes"
ID="appearance_and_display\windows_games_settings">
        <component displayname="Microsoft-Windows-GameExplorer"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
gameexplorer/microsoft-windows-gameexplorer/settings"/>
      </component>
      <component displayname="Taskbar and Start Menu" migrate="yes"
ID="appearance_and_display\taskbar_and_start_menu">
        <component displayname="Microsoft-Windows-stobject" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
stobject/microsoft-windows-stobject/settings"/>
        <component displayname="Microsoft-Windows-explorer" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
explorer/microsoft-windows-explorer/settings"/>
      </component>
      <component displayname="Personalized Settings" migrate="yes"
ID="appearance_and_display\personalized_settings">
        <component displayname="Microsoft-Windows-uxtheme" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
uxtheme/microsoft-windows-uxtheme/settings"/>
        <component displayname="Microsoft-Windows-themeui" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
themeui/microsoft-windows-themeui/settings"/>
        <component displayname="Microsoft-Windows-shmig" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
shmig/microsoft-windows-shmig/settings"/>
        <component displayname="Microsoft-Windows-shell32" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
shell32/microsoft-windows-shell32/settings"/>
        <component displayname="Microsoft-Windows-CommandPrompt"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
commandprompt/microsoft-windows-commandprompt/settings"/>
      </component>
    </component>
    <component displayname="Additional Options" migrate="yes"
ID="additional_options">
      <component displayname="Help Settings" migrate="yes"
ID="additional_options\help_settings">
        <component displayname="Microsoft-Windows-Help-Client" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
help-client/microsoft-windows-help-client/settings"/>
      </component>
      <component displayname="Windows Core Settings" migrate="yes"
ID="additional_options\windows_core_settings">
        <component displayname="Microsoft-Windows-Win32k-Settings"
migrate="yes"
```

```
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
win32k-settings/microsoft-windows-win32k-settings/settings"/>
        <component displayname="Microsoft-Windows-Web-Services-for-
Management-Core" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
web-services-for-management-core/microsoft-windows-web-services-for-
management-core/settings"/>
        <component displayname="Microsoft-Windows-UPnPSSDP" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
upnpssdp/microsoft-windows-upnpssdp/settings"/>
        <component displayname="Microsoft-Windows-UPnPDeviceHost"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
upnpdevicehost/microsoft-windows-upnpdevicehost/settings"/>
        <component displayname="Microsoft-Windows-UPnPControlPoint"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
upnpcontrolpoint/microsoft-windows-upnpcontrolpoint/settings"/>
        <component displayname="Microsoft-Windows-TerminalServices-
RemoteConnectionManager" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
terminalservices-remoteconnectionmanager/microsoft-windows-terminalservices-
remoteconnectionmanager/settings"/>
        <component displayname="Microsoft-Windows-TerminalServices-Drivers"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
terminalservices-drivers/microsoft-windows-terminalservices-
drivers/settings"/>
        <component displayname="Microsoft-Windows-SQMApi" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
sqmapi/microsoft-windows-sqmapi/settings"/>
        <component displayname="Microsoft-Windows-RPC-Remote" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
rpc-remote/microsoft-windows-rpc-remote/settings"/>
        <component displayname="Microsoft-Windows-RPC-Local" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
rpc-local/microsoft-windows-rpc-local/settings"/>
        <component displayname="Microsoft-Windows-RPC-HTTP" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
rpc-http/microsoft-windows-rpc-http/settings"/>
        <component displayname="Microsoft-Windows-Rasppp" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
rasppp/microsoft-windows-rasppp/settings"/>
        <component displayname="Microsoft-Windows-RasMprDdm" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
rasmprddm/microsoft-windows-rasmprddm/settings"/>
        <component displayname="Microsoft-Windows-RasBase" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
rasbase/microsoft-windows-rasbase/settings"/>
        <component displayname="Microsoft-Windows-Microsoft-Data-Access-
Components-(MDAC)-ODBC-DriverManager-Dll" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
microsoft-data-access-components-(mdac)-odbc-drivermanager-dll/microsoft-
windows-microsoft-data-access-components-(mdac)-odbc-drivermanager-
dll/settings"/>
```

```
        <component displayname="Microsoft-Windows-ICM-Profiles"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
icm-profiles/microsoft-windows-icm-profiles/settings"/>
        <component displayname="Microsoft-Windows-feclient" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
feclient/microsoft-windows-feclient/settings"/>
        <component displayname="Microsoft-Windows-dpapi-keys" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
dpapi-keys/microsoft-windows-dpapi-keys/settings"/>
        <component displayname="Microsoft-Windows-Crypto-keys" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
crypto-keys/microsoft-windows-crypto-keys/settings"/>
        <component displayname="Microsoft-Windows-COM-DTC-Setup"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
com-dtc-setup/microsoft-windows-com-dtc-setup/settings"/>
        <component displayname="Microsoft-Windows-COM-ComPlus-Setup"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
com-complus-setup/microsoft-windows-com-complus-setup/settings"/>
        <component displayname="Microsoft-Windows-COM-Base" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
com-base/microsoft-windows-com-base/settings"/>
        <component displayname="Microsoft-Windows-CAPI2-certs" migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
capi2-certs/microsoft-windows-capi2-certs/settings"/>
      </component>
    </component>
    <component displayname="Accessibility" migrate="yes" ID="accessibility">
      <component displayname="Accessibility Settings" migrate="yes"
ID="accessibility\accessibility_settings">
        <component displayname="Microsoft-Windows-accessibilitycpl"
migrate="yes"
ID="http://www.microsoft.com/migration/1.0/migxmlext/cmi/microsoft-windows-
accessibilitycpl/microsoft-windows-accessibilitycpl/settings"/>
      </component>
    </component>
  </WindowsComponents>
  <Policies>
    <ErrorControl>
      <!--   Example:

          <fileError>
            <nonFatal errorCode="33">* [*]</nonFatal>
            <fatal errorCode="any">C:\Users\* [*]</fatal>
          </fileError>
          <registryError>
            <nonFatal errorCode="5">* [*]</nonFatal>
          </registryError>
      -->
    </ErrorControl>
    <HardLinkStoreControl>
      <!--   Example:
```

```
          <fileLocked>
            <createHardLink>c:\Users\* [*]</createHardLink>
            <errorHardLink>C:\* [*]</errorHardLink>
          </fileLocked>
      -->
    </HardLinkStoreControl>
  </Policies>
  <ProfileControl>
    <!--   Example:

          <localGroups>
            <mappings>
              <changeGroup from="Administrators" to="Users"
appliesTo="MigratedUsers">
                <include>
                  <pattern>DomainName1\Username</pattern>
                </include>
                <exclude>
                  <pattern>DomainName2\Username</pattern>
                </exclude>
              </changeGroup>
            </mappings>
          </localGroups>

      -->
  </ProfileControl>
</Configuration>
```

Related articles

- USMT XML reference.

Feedback

Was this page helpful? Yes No

Provide product feedback

Customize USMT XML files

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

Overview

To use any of the migration **.xml** files with the **ScanState** and **LoadState** tools, specify these files at the command line using the `/i` option. Because the **ScanState** and **LoadState** tools need the **.xml** files to control the migration, specify the same set of **.xml** files for both the `ScanState.exe` and `LoadState.exe` commands. However, the `Config.xml` file with the `/config` option doesn't need to be specified, unless some of the migrated files and settings from the store need to be excluded. For example, to migrate the **Documents** folder to the store but not to the destination computer. To achieve this scenario, modify the `Config.xml` file and specify the updated file with the `LoadState.exe` command. The `LoadState.exe` command then only migrates the desired files and settings.

If an **.xml** file is left out from the `LoadState.exe` command, all of the data in the store that was migrated with the missing **.xml** files are migrated. However, the migration rules that were specified with the `ScanState.exe` command don't apply. For example, if an **.xml** file is left out, and it contains a rerouting rule such as:

```
MigsysHelperFunction.RelativeMove("c:\data", "%CSIDL_PERSONAL%")
```

USMT doesn't reroute the files, and they're migrated to `C:\data`.

To modify the migration, do one or more of the following.

- **Modify the migration .xml files.** To exclude a portion of a component, modify the **.xml** files. For example, to migrate C:\ but exclude all of the **.mp3** files, or to move data to a new location on the destination computer. To modify these files, familiarity with the migration rules and syntax is a must. For **ScanState** and **LoadState** to use these files, specify them at the command line when each command is entered.

- **Create a custom .xml file.** A custom **.xml** file can also be created to migrate settings for another application, or to change the migration behavior to suit the organization's needs. For **ScanState** and **LoadState** to use this file, specify them on both command lines.

- **Create and modify a Config.xml file.** Create and modify a `Config.xml` file to exclude an entire component from the migration. For example, a `Config.xml` file

can be used to exclude the entire **Documents** folder, or exclude the settings for an application. Excluding components using a `Config.xml` file is easier than modifying the migration **.xml** files because familiarity with the migration rules and syntax isn't needed. In addition, using a `Config.xml` file is the only way to exclude the operating system settings from being migrated.

For more information about excluding data, see the Exclude Files and Settings article.

Migration .xml files

This section describes the migration **.xml** files that are included with USMT. Each file contains migration rules that control which components are migrated and where they're migrated to on the destination computer.

> **Note**
>
> The asterisk (*) wildcard character can be used in each of these files. However, a question mark (?) can't be used as a wildcard character.

- **The MigApp.xml file.** Specify this file with both the `ScanState.exe` and `LoadState.exe` commands to migrate application settings.
- **The MigDocs.xml file.** Specify this file with both the **ScanState** and **LoadState** tools to migrate all user folders and files that are found by the **MigXmlHelper.GenerateDocPatterns** helper function. This helper function finds user data that resides on the root of any drive and in the Users directory. However, it doesn't find and migrate any application data, program files, or any files in the Windows directory. The `MigDocs.xml` file can be modified.
- **The MigUser.xml file.** Specify this file with both the `ScanState.exe` and `LoadState.exe` commands to migrate user folders, files, and file types. The `MigUser.xml` file can be modified. This file doesn't contain rules that migrate specific user accounts. The only way to specify which user accounts to migrate is on the command line by using the ScanState User options and the LoadState User options.

> **Note**
>
> Don't use the `MigUser.xml` and `MigDocs.xml` files together. For more information, see the Identify file types, files, and folders and USMT best practices articles.

Custom .xml files

Custom **.xml** files can be created to customize the migration for the organization's unique needs. For example, a custom **.xml** file can be created to migrate a line-of-business application or to modify the default migration behavior. For `ScanState.exe` and `LoadState.exe` to use this file, specify it with both commands. For more information, see the Custom XML examples article.

The Config.xml file

The `Config.xml` file is an optional file that is created using the `/genconfig` option with the `ScanState.exe` command. This file should be created and modified to exclude certain components from the migration. In addition, this file must be created and modified to exclude any of the operating system settings from being migrated. The `Config.xml` file format is different from the migration **.xml** files because it doesn't contain any migration rules. It contains only a list of the operating system components, applications, and the user documents that can be migrated. For an example, see the Config.xml File article. For this reason, excluding components using the `Config.xml` file is easier than modifying the migration **.xml** files. With the `Config.xml`, familiarity with the migration rules and syntax isn't. However, wildcard characters can't be used in a `Config.xml` file.

To include all of the default components, a `Config.xml` file doesn't need to be created. Alternatively, if the default migration behavior defined in the `MigApp.xml`, `MigDocs.xml`, and `MigUser.xml` files are satisfactory, and only some components need to be excluded, a `Config.xml` file can be created. The other **.xml** files can be left in their original state.

When the `ScanState.exe` command is run with the `/genconfig` option, `ScanState.exe` reads the other **.xml** files that are specified using the `/i` option to create a custom list of components that can be migrated from the computer. This file contains only operating system components, applications, and the user document sections that are in both of the **.xml** files and that are installed on the computer when the `ScanState.exe` command is run with the `/genconfig` option. Therefore, this file should be created on a source computer that contains all of the components, applications, and settings that are present on the destination computers. Creating the file on the source computer ensures that this file contains every component that can be migrated. The components are organized into sections: <Applications>, <WindowsComponents>, and <Documents>. To choose not to migrate a component, change its entry to `migrate="no"`.

After this file is created, it only needs to be specified with the `ScanState.exe` command using the `/Config` option for it to affect the migration. However, if additional data that was migrated to the store needs to be excluded, modify the `Config.xml` file and specify the updated file with the `LoadState.exe` command. For example, if the **Documents** folder is collected in the store, but the **Documents** folder doesn't need to be migrated to a destination computer, the `Config.xml` file can be modified to indicate `migrate="no"` before the `LoadState.exe` command runs, and the file aren't be migrated. For more information about the precedence that takes place when excluding data, see the Exclude files and settings article.

In addition, note the following functionality with the `Config.xml` file:

- If a parent component is removed from the migration in the `Config.xml` file by specifying `migrate="no"`, all of its child components are automatically removed from the migration, even if the child component is set to `migrate="yes"`.
- If mistakenly two lines of code exist for the same component where one line specifies `migrate="no"` and the other line specifies `migrate="yes"`, the component is migrated.
- In USMT, there are several migration policies that can be configured in the `Config.xml` file. For example, additional **<ErrorControl>**, **<ProfileControl>**, and **<HardLinkStoreControl>** options can be configured. For more information, see the Config.xml File article.

> **Note**
>
> To exclude a component from the `Config.xml` file, set the **migrate** value to **"no"**. Deleting the XML tag for the component from the `Config.xml` file doesn't exclude the component from the migration.

Examples

- The following command creates a `Config.xml` file in the current directory, but it doesn't create a store:

  ```
  ScanState.exe /i:MigApp.xml /i:MigDocs.xml /genconfig:Config.xml /v:5
  ```

- The following command creates an encrypted store using the `Config.xml` file and the default migration **.xml** files:

```
ScanState.exe \\server\share\migration\mystore /i:MigApp.xml /i:MigDocs.xml /o
/config:Config.xml /v:5 /encrypt /key:"mykey"
```

- The following command decrypts the store and migrates the files and settings:

```
LoadState.exe \\server\share\migration\mystore /i:MigApp.xml /i:MigDocs.xml
/v:5 /decrypt /key:"mykey"
```

Additional information

- For more information about how to change the files and settings that are migrated, see the User State Migration Tool (USMT) how-to articles.
- For more information about each **.xml** element, see the XML elements library article.
- For answers to common questions, see ".xml files" in the Frequently asked questions article.

Related articles

- User State Migration Tool (USMT) command-line syntax.
- USMT resources.

Feedback

Was this page helpful? Yes No

Provide product feedback

Custom XML Examples

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

Example 1: Migrating an unsupported application

The following template is a template for the sections that are needed to migrate applications. The template isn't functional on its own, but it can be used to write custom **.xml** file.

Template

▼ Expand to show **Example 1** application template:

XML

```
<migration
urlid="http://www.microsoft.com/migration/1.0/migxmlext/migtestapp">
  <component type="Application">
    <!-- Name of the application -->
    <displayName>Some Application</displayName>
    <!-- Specify whether the environment variables exist in the context of
user or system or both -->
    <environment context="System">
      <!-- Create the environment variables -->
      <variable name="myVar1">
        <!-- Simple text value assignment to a variable -->
        <text>value</text>
      </variable>
      <variable name="myAppExePath">
        <!-- Make a call to in-built helper function to get a value from a
reg key and assign that value to the variable -->

<script>MigXMLHelper.GetStringContent("Registry","HKLM\Software\MyApp\Instal
ler [EXEPATH]")</script>
      </variable>
    </environment>
    <role role="Settings">
      <detects>
        <!-- All of these checks must be true for the component to be
detected -->
        <detect>
          <!-- Make a call to in-built helper function to check if an object
exists or not -->

<condition>MigXMLHelper.DoesObjectExist("Registry","HKLM\Software\MyApp
[win32_version]")</condition>
        </detect>
        <detect>
```

```
          <!-- Either of these checks must be true for the component to be
detected -->
          <!-- Make a call to in-built helper function to check if a file
version matches or not -->

<condition>MigXMLHelper.DoesFileVersionMatch("%MyAppExePath%","ProductVersio
n","8.*")</condition>

<condition>MigXMLHelper.DoesFileVersionMatch("%MyAppExePath%","ProductVersio
n","9.*")</condition>
        </detect>
      </detects>
      <!-- Describe the rules that will be executed during migration of this
component and the context, whether user, system or both -->
      <rules context="User">
        <!-- Delete objects specified in the object set on the destination
computer before applying source objects -->
        <destinationCleanup>
          <!-- Describe the pattern for the list of objects to be deleted --
>
          <objectSet>
            <pattern type="Registry">HKCU\Software\MyApp\Toolbar\* [*]
</pattern>
            <pattern type="Registry">HKCU\Software\MyApp\ListView\* [*]
</pattern>
            <pattern type="Registry">HKCU\Software\MyApp [ShowTips]
</pattern>
          </objectSet>
        </destinationCleanup>
        <!-- Specify which set of objects should be migrated -->
        <include>
          <!-- Describe the pattern for the list of objects to be included -
->
          <objectSet>
            <pattern type="Registry">HKCU\Software\MyApp\Toolbar\* [*]
</pattern>
            <pattern type="Registry">HKCU\Software\MyApp\ListView\* [*]
</pattern>
            <pattern type="Registry">HKCU\Software\MyApp [ShowTips]
</pattern>
          </objectSet>
        </include>
        <!-- Specify which set of objects should not be migrated -->
        <exclude>
          <!-- Describe the pattern for the list of objects to be excluded
from migration -->
          <objectSet>
            <pattern type="Registry">HKCU\Software\MyApp [Display]</pattern>
          </objectSet>
        </exclude>
      </rules>
    </role>
  </component>
</migration>
```

Example 2: Migrating the My Videos folder

The following sample is a custom **.xml** file named `CustomFile.xml` that migrates the **Videos** folder for all users, if the folder exists on the source computer.

- **Sample condition**: Verifies that the **Videos** folder exists on the source computer:

  `<condition>MigXmlHelper.DoesObjectExist("File","%CSIDL_MYVIDEO%")</condition>`

- **Sample filter**: Filters out the shortcuts in the **Videos** folder that don't resolve on the destination computer:

  `<include filter='MigXmlHelper.IgnoreIrrelevantLinks()'>`

  This filter has no effect on files that aren't shortcuts. For example, if there's a shortcut in the **Videos** folder on the source computer that points to `C:\Folder1`, that shortcut is migrated only if `C:\Folder1` exists on the destination computer. However, all other files, such as **.mp3** files, migrate without any filtering.

- **Sample pattern**: Migrates the **Videos** folder for all users:

  `<pattern type="File">%CSIDL_MYVIDEO%* [*]</pattern>`

XML file

▼ Expand to show **Example 2** XML file:

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<migration
urlid="http://www.microsoft.com/migration/1.0/migxmlext/CustomFile">
<component type="Documents" context="User">
        <displayName>My Video</displayName>
        <role role="Data">
            <detects>
                <detect>

<condition>MigXmlHelper.DoesObjectExist("File","%CSIDL_MYVIDEO%")
</condition>
                </detect>
            </detects>
            <rules>
                <include filter='MigXmlHelper.IgnoreIrrelevantLinks()'>
                    <objectSet>
                        <pattern type="File">%CSIDL_MYVIDEO%\* [*]</pattern>
                    </objectSet>
                </include>
            </rules>
        </role>
```

```
    </component>
</migration>
```

Example 3: Migrating files and registry keys

The sample patterns describe the behavior in the following example **.xml** file.

- **Sample pattern**: Migrates all instances of the file `Usmttestfile.txt` from all subdirectories under `%ProgramFiles%\USMTTestFolder`:

  `<pattern type="File">%ProgramFiles%\USMTTestFolder* [USMTTestFile.txt] </pattern>`

- **Sample pattern**: Migrates the whole directory under `%ProgramFiles%\USMTDIRTestFolder`:

  `<pattern type="File">%ProgramFiles%\USMTDIRTestFolder* []</pattern>`

- **Sample pattern**: Migrates all instances of MyKey under `HKCU\Software\USMTTESTKEY`:

  `<pattern type="Registry">HKCU\Software\USMTTESTKEY* [MyKey]</pattern>`

- **Sample pattern**: Migrates the entire registry hive under `HKLM\Software\USMTTESTKEY`:

  `<pattern type="Registry">HKLM\Software\USMTTESTKEY* []</pattern>`

XML file

▼ Expand to show **Example 3** XML file:

XML

```
<migration
urlid="http://www.microsoft.com/migration/1.0/migxmlext/testfilemig">
  <component type="Application" context="System">
   <displayName>File Migration Test</displayName>
   <role role="Data">
    <rules context="System">
     <include>
      <objectSet>
        <pattern type="File">%ProgramFiles%\USMTTestFolder\*
[USMTTestFile.txt]</pattern>
        <pattern type="File">%ProgramFiles%\USMTDIRTestFolder\* [*]
</pattern>
      </objectSet>
    </include>
```

```
    </rules>
   </role>
  </component>
  <component type="System">
   <displayName>Registry Migration Test</displayName>
   <role role="Settings">
    <rules context="UserAndSystem">
      <include>
       <objectSet>
           <pattern type="Registry">HKCU\Software\USMTTESTKEY\* [MyKey]
</pattern>
           <pattern type="Registry">HKLM\Software\USMTTESTKEY\* [*]</pattern>
       </objectSet>
      </include>
    </rules>
   </role>
  </component>
</migration>
```

Example 4: Migrating specific folders from various locations

The behavior for this custom **.xml** file is described within the `<displayName>` tags in the code.

XML file

▼ Expand to show **Example 4** XML file:

XML

```
<migration urlid="http://www.microsoft.com/migration/1.0/migxmlext/test">

<component type="Documents" context="System">
  <displayName>Component to migrate all Engineering Drafts subfolders
without documents in this folder </displayName>
  <role role="Data">
    <rules>
         <include>
            <objectSet>
                 <pattern type="File"> C:\EngineeringDrafts\* [*]</pattern>
            </objectSet>
          </include>
         <exclude>
            <objectSet>
                 <pattern type="File"> C:\EngineeringDrafts\ [*]</pattern>
            </objectSet>
         </exclude>
    </rules>
  </role>
</component>
```

```
<component type="Documents" context="System">
  <displayName>Component to migrate all user documents except
Sample.doc</displayName>
  <role role="Data">
    <rules>
          <include>
            <objectSet>
                 <pattern type="File"> C:\UserDocuments\* [*]</pattern>
            </objectSet>
          </include>
          <exclude>
             <objectSet>
                 <pattern type="File"> C:\UserDocuments\ [Sample.doc]
</pattern>
             </objectSet>
          </exclude>
    </rules>
  </role>
</component>

<component type="Documents" context="System">
  <displayName>Component to migrate all Requests folders on any drive on the
computer </displayName>
  <role role="Data">
    <rules>
         <include>
            <objectSet>
                 <script>MigXmlHelper.GenerateDrivePatterns ("\Requests\*
[*] ", "Fixed")</script>
                 <script>MigXmlHelper.GenerateDrivePatterns ("*\Requests\*
[*] ", "Fixed")</script>
            </objectSet>
          </include>
    </rules>
  </role>
</component>

<component type="Documents" context="System">
  <displayName>Component to migrate all Presentations folder from any
location on the C: drive </displayName>
  <role role="Data">
    <rules>
         <include>
            <objectSet>
                 <pattern type="File"> C:\*\Presentations\* [*]</pattern>
                 <pattern type="File"> C:\Presentations\* [*]</pattern>
            </objectSet>
          </include>
    </rules>
  </role>
</component>
</migration>
```

Related articles

- USMT XML reference.
- Customize USMT XML files.

Feedback

Was this page helpful? Yes No

Provide product feedback

Conflicts and precedence

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

When including, excluding, and rerouting files and settings, it's important to know how User State Migration Tool (USMT) deals with conflicts and precedence. The following are the most important conflicts and precedence guidelines to keep in mind when working with USMT.

- **If there are conflicting rules within a component, the most specific rule is applied.** However, the **<unconditionalExclude>** rule is an exception because it takes precedence over all others. Directory names take precedence over file extensions. For examples, see What happens when there are conflicting <include> and <exclude> rules? and the first example in <include> and <exclude> rules precedence examples later in this article.
- **Only rules inside the same component can affect each other, depending on specificity.** Rules that are in different components don't affect each other, except for the **<unconditionalExclude>** rule.
- **If the rules are equally specific, <exclude> takes precedence over <include>.** For example, if the **<exclude>** rule is used to exclude a file and use the **<include>** rule to include the same file, the file is excluded.
- **The ordering of components does not matter.** It doesn't matter which components are listed in which **.xml** file, because each component is processed independently of the other components across all of the **.xml** files.
- **The ordering of the <include> and <exclude> rules within a component does not matter.**
- **The <unconditionalExclude> element can be used to globally exclude data.** This element excludes objects, regardless of any other **<include>** rules that are in the **.xml** files. For example, the **<unconditionalExclude>** element can be used to exclude all MP3 files on the computer or to exclude all files from `C:\UserData`.

General

What is the relationship between rules that are located within different components?

Only rules inside the same component can affect each other, depending on specificity, except for the **<unconditionalExclude>** rule. Rules that are in different components don't affect each other. If there's an **<include>** rule in one component and an identical **<exclude>**

rule in another component, the data is migrated because the two rules are independent of each other.

If an **<include>** rule is in one component and a **<locationModify>** rule is in another component for the same file, the file is migrated in both places. That is, the file is included based on the **<include>** rule, and the file is migrated based on the **<locationModify>** rule.

The following **.xml** file migrates all files from C:\Userdocs, including **.mp3** files, because the **<exclude>** rule is specified in a separate component.

XML

```xml
<migration urlid="http://www.microsoft.com/migration/1.0/migxmlext/UserDocs">
<component type="Documents" context="System">
<displayName>User Documents</displayName>
        <role role="Data">
            <rules>
                <exclude>
                    <objectSet>
                        <pattern type="File">C:\Userdocs\* [*.mp3]</pattern>
                    </objectSet>
                </exclude>
          </rules>
        </role>
</component>

<component type="Documents" context="System">
<displayName> User documents to include </displayName>
        <role role="Data">
            <rules>
                <include>
                    <objectSet>
                        <pattern type="File"> C:\Userdocs\ [*]</pattern>
                    </objectSet>
                </include>
          </rules>
        </role>
</component>
</migration>
```

How does precedence work with the Config.xml file?

Specifying `migrate="no"` in the `Config.xml` file is the same as deleting the corresponding component from the migration **.xml** file. However, if `migrate="no"` is set for the **Documents** folder, but a rule similar to the following rule exists in a migration **.xml** file (which includes all of the **.doc** files from the **Documents** folder), then only the **.doc** files is migrated, and all other files are excluded:

XML

```
<include>
   <objectSet>
      <pattern type="File">%CSIDL_PERSONAL%\* [*.doc] </pattern>
   </objectSet>
</include>
```

How does USMT process each component in an .xml file with multiple components?

The ordering of components doesn't matter. Each component is processed independently of other components. For example, if an **<include>** rule is in one component and a **<locationModify>** rule is in another component for the same file, the file is migrated in both places. That is, the file is included based on the **<include>** rule, and the file is migrated based on the **<locationModify>** rule.

How are rules processed?

There are two broad categories of rules.

- **Rules that affect the behavior of both the ScanState and LoadState tools**. For example, the **<include>**, **<exclude>**, and **<unconditionalExclude>** rules are processed for each component in the **.xml** files. For each component, USMT creates an include list and an exclude list. Some of the rules in the component might be discarded due to specificity, but all of the remaining rules are processed. For each **<include>** rule, USMT iterates through the elements to see if any of the locations need to be excluded. USMT enumerates all of the objects and creates a list of objects it's going to collect for each user. Once the list is complete, each of the objects is stored or migrated to the destination computer.

- **Rules that affect the behavior of only the LoadState tool**. For example, the **<locationModify>**, **<contentModify>**, and **<destinationCleanup>** rules don't affect ScanState. They're processed only with LoadState. First, the **LoadState** tool determines the content and location of each component based on the **<locationModify>** and **<contentModify>** rules. Then, **LoadState** processes all of the **<destinationCleanup>** rules and deletes data from the destination computer. Lastly, **LoadState** applies the components to the computer.

How does USMT combine all of the .xml files that I specify on the command line?

USMT doesn't distinguish the **.xml** files based on their name or content. It processes each component within the files separately. USMT supports multiple **.xml** files only to make it

easier to maintain and organize the components within them. Because USMT uses a urlid to distinguish each component from the others, be sure that each **.xml** file that is specified on the command line has a unique migration urlid.

The <include> and <exclude> rules

What happens when there are conflicting <include> and <exclude> rules?

If there are conflicting rules within a component, the most specific rule is applied, except with the **<unconditionalExclude>** rule, which takes precedence over all other rules. If the rules are equally specific, then the data isn't migrated. For example if the same file is both excluded and included, the file isn't migrated. If there are conflicting rules within different components, the rules don't affect each other because each component is processed independently.

In the following example, mp3 files aren't excluded from the migration. The mp3 files aren't excluded because directory names take precedence over the file extensions.

XML

```xml
<include>
     <objectSet>
          <pattern type="File">C:\Data\* [*]</pattern>
     </objectSet>
</include>
<exclude>
     <objectSet>
          <pattern type="File"> C:\* [*.mp3]</pattern>
     </objectSet>
</exclude>
```

<include> and <exclude> rules precedence examples

These examples explain how USMT deals with **<include>** and **<exclude>** rules. When the rules are in different components, the resulting behavior is the same regardless of whether the components are in the same or in different migration **.xml** files.

- Including and excluding files
- Including and excluding registry objects

Including and excluding files

Expand table

If the following code exists in the same component	Resulting behavior	Explanation
• Include rule: <pattern type="File">C:\Dir1* [] </pattern> • Exclude rule: <pattern type="File">C:* [.txt]</pattern>	Migrates all files and subfolders in Dir1 (including all **.txt** files in C:).	The **<exclude>** rule doesn't affect the migration because the **<include>** rule is more specific.
• Include rule: <pattern type="File">C:\Dir1* [] </pattern> • Exclude rule: <pattern type="File">C:\Dir1\Dir2* [.txt] </pattern>	Migrates all files and subfolders in C:\Dir1, except the **.txt** files in C:\Dir1\Dir2 and its subfolders.	Both rules are processed as intended.
• Include rule: <pattern type="File">C:\Dir1* [] </pattern> • Exclude rule: <pattern type="File">C:\Dir1\ * [.txt] </pattern>	Migrates all files and subfolders in C:\Dir1, except the **.txt** files in C:\Dir1 and its subfolders.	Both rules are processed as intended.
• Include rule: <pattern type="File">C:\Dir1\Dir2* [.txt] </pattern> • Exclude rule: <pattern type="File">C:\Dir1\Dir2* [.txt] </pattern>	Nothing is migrated.	The rules are equally specific, so the **<exclude>** rule takes precedence over the **<include>** rule.
• Include rule: C:\Dir1* [.txt] • Exclude rule: C:\Dir1\Dir2* []	Migrates the **.txt** files in Dir1 and the **.txt** files from subfolders other than Dir2. No files are migrated from Dir2 or its subfolders.	Both rules are processed as intended.
• Include rule: C:\Dir1\Dir2* [] • Exclude rule: C:\Dir1* [.txt]	Migrates all files and subfolders of Dir2, except the **.txt** files from Dir1 and any subfolders of Dir1 (including Dir2).	Both rules are processed as intended.

Expand table

If the following code exists in different components	Resulting behavior	Explanation
Component 1: • Include rule: <pattern type="File">C:\Dir1* [] </pattern> • Exclude rule: <pattern type="File">C:\Dir1\Dir2* [.txt]</pattern> Component 2: • Include rule: <pattern type="File">C:\Dir1\Dir2* [.txt]</pattern> • Exclude rule: <pattern type="File">C:\Dir1* [] </pattern>	Migrates all files and subfolders of C:\Dir1\ (including C:\Dir1\Dir2).	Rules that are in different components don't affect each other, except for the **<unconditionalExclude>** rule. Therefore, in this example, although some **.txt** files were excluded when Component 1 was processed, they were included when Component 2 was processed.
Component 1: • Include rule: C:\Dir1\Dir2* [] Component 2: • Exclude rule: C:\Dir1* [.txt]	Migrates all files and subfolders from Dir2 except the **.txt** files in C:\Dir1 and its subfolders.	Both rules are processed as intended.
Component 1: • Exclude rule: C:\Dir1\Dir2* [] Component 2: • Include rule: C:\Dir1* [.txt]	Migrates all **.txt** files in Dir1 and any subfolders.	Component 1 doesn't contain an **<include>** rule, so the **<exclude>** rule isn't processed.

Including and excluding registry objects

Expand table

If the following code exists in the same component	Resulting behavior	Explanation
• Include rule: HKLM\Software\Microsoft\Command Processor* [] • Exclude Rule: HKLM\Software\Microsoft\Command	Migrates all keys in HKLM\Software\Microsoft\Command Processor except DefaultColor.	Both rules are processed as intended.

If the following code exists in the same component	Resulting behavior	Explanation
Processor [DefaultColor]		
• Include rule: HKLM\Software\Microsoft\Command Processor [DefaultColor] • Exclude Rule: HKLM\Software\Microsoft\Command Processor* []	Migrates only DefaultColor in HKLM\Software\Microsoft\Command Processor.	DefaultColor is migrated because the **<include>** rule is more specific than the **<exclude>** rule.
• Include rule: HKLM\Software\Microsoft\Command Processor [DefaultColor] • Exclude rule: HKLM\Software\Microsoft\Command Processor [DefaultColor]	Doesn't migrate DefaultColor.	The rules are equally specific, so the **<exclude>** rule takes precedence over the <include> rule.

Expand table

If the following code exists in different components	Resulting behavior	Explanation
Component 1: • Include rule: HKLM\Software\Microsoft\Command Processor [DefaultColor] • Exclude rule: HKLM\Software\Microsoft\Command Processor* [] Component 2: • Include rule: HKLM\Software\Microsoft\Command Processor* [] • Exclude rule: HKLM\Software\Microsoft\Command Processor [DefaultColor]	Migrates all the keys/values under HKLM\Software\Microsoft\Command Processor.	Rules that are in different components don't affect each other, except for the **<unconditionalExclude>** rule. In this example, the objects that were excluded when Component 1 was processed were included when Component 2 was processed.

File collisions

What is the default behavior when there are file collisions?

If there isn't a **<merge>** rule, the default behavior for the registry is for the source to overwrite the destination. The default behavior for files is for the source to be renamed incrementally: for example, OriginalFileName(1).OriginalExtension, OriginalFileName(2).OriginalExtension, and so on.

How does the <merge> rule work when there are file collisions?

When a collision is detected, USMT selects the most specific **<merge>** rule and apply it to resolve the conflict. For example, if a **<merge>** rule exists for **C:* [*]** set to **sourcePriority()** and another **<merge>** rule for **C:\subfolder* [*]** set to **destinationPriority()** , then USMT uses the **destinationPriority()** rule because it's the most specific.

Example scenario

The source computer contains the following files:

- `C:\Data\SampleA.txt`
- `C:\Data\SampleB.txt`
- `C:\Data\Folder\SampleB.txt`

The destination computer contains the following files:

- `C:\Data\SampleB.txt`
- `C:\Data\SampleB.txt`

A custom **.xml** file contains the following code:

```xml
<include>
   <objectSet>
      <pattern type="File">c:\data\* [*]</pattern>
   </objectSet>
</include>
```

For this example, the following information describes the resulting behavior if the code is added to the custom **.xml** file.

Example 1

```XML
<merge script="MigXmlHelper.DestinationPriority()">
        <objectSet>
                <pattern type="File">c:\data* []</pattern>
        </objectSet>
</merge>
```

Result: During ScanState, all the files are added to the store. During LoadState, only `C:\Data\SampleA.txt` is restored.

Example 2

```XML
<merge script="MigXmlHelper.SourcePriority()">
        <objectSet>
                <pattern type="File">c:\data* []</pattern>
        </objectSet>
</merge>
```

Result: During ScanState, all the files are added to the store. During LoadState, all the files are restored, overwriting the existing files on the destination computer.

Example 3

```XML
<merge script="MigXmlHelper.SourcePriority()">
        <objectSet>
                <pattern type="File">c:\data\ [*]</pattern>
        </objectSet>
</merge>
```

Result: During ScanState, all the files are added to the store. During LoadState, the following actions occur:

- `C:\Data\SampleA.txt` is restored.
- `C:\Data\SampleB.txt` is restored, overwriting the existing file on the destination computer.
- `C:\Data\Folder\SampleB.txt` aren't restored.

Related articles

USMT XML reference.

Feedback

Was this page helpful? Yes No

Provide product feedback

General conventions

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

This article describes the XML helper functions.

General XML guidelines

Before modifying the **.xml** files, become familiar with the following guidelines:

- **XML schema.**

  The User State Migration Tool (USMT) XML schema, `MigXML.xsd`, can be used to write and validate migration **.xml** files.

- **Conflicts.**

  In general, when there are conflicts within the XML schema, the most specific pattern takes precedence. For more information, see Conflicts and precedence.

- **Required elements.**

  The required elements for a migration **.xml** file are **<migration>**, **<component>**, **<role>**, and **<rules>**.

- **Required child elements.**

  - USMT doesn't fail with an error if the required child elements aren't specified. However, the required child elements must be specified for the parent element to affect the migration.

  - The required child elements apply only to the first definition of the element. If these elements are defined and then referred to using their name, the required child elements don't apply. For example, if `<detects name="Example">` is defined in **<namedElements>**, and `<detects name="Example"/>` is specified in **<component>** to refer to this element, the definition inside **<namedElements>** must have the required child elements, but the **<component>** element doesn't need to have the required child elements.

- **File names with brackets.**

  If a file that has a bracket character ([or]) in the file name is being migrated, a carat (^) character must be inserted. The carat (^) character must be directly before the bracket for the bracket character to be valid. For example, if there's a

file named **file].txt**, `<pattern type="File">c:\documents\mydocs [file^].txt]</pattern>` must be specified instead of `<pattern type="File">c:\documents\mydocs [file].txt]</pattern>`.

- **Using quotation marks.**

  When code is surrounded in quotation marks, either the double ("") or the single (') quotation marks can be used.

Helper functions

The XML helper functions in the XML elements library can be used to change migration behavior. Before using these functions in an **.xml** file, note the following items:

- **All of the parameters are strings.**

- **NULL parameters can be left blank.**

  As with parameters with a default value convention, if there's a NULL parameter at the end of a list, it can be left out. For example, the following function:

  Windows Command Prompt

  ```
  SomeFunction("My String argument",NULL,NULL)
  ```

  is equivalent to:

  Windows Command Prompt

  ```
  SomeFunction("My String argument")
  ```

- **The encoded location used in all the helper functions is an unambiguous string representation for the name of an object.**

The encoded location is composed of the node part, optionally followed by the leaf enclosed in square brackets. This format makes a clear distinction between nodes and leaves.

For example, specify the file

`C:\Windows\Notepad.exe`

as

c:\Windows[Notepad.exe]

Similarly, specify the directory

`C:\Windows\System32`

as

c:\Windows\System32

Note the absence of the **[]** characters in second example.

The registry is represented in a similar way. The default value of a registry key is represented as an empty **[]** construct. For example, the default value for the `HKLM\SOFTWARE\MyKey` registry key is **HKLM\SOFTWARE\MyKey[]**.

- **A location pattern is specified in a way that is similar to how an actual location is specified.**

  The exception is that both the node and leaf part accept patterns. However, a pattern from the node doesn't extend to the leaf.

  For example, the pattern **c:\Windows*** matches the `\Windows` directory and all subdirectories, but it doesn't match any of the files in those directories. To match the files as well, **c:\Windows*[*]** must be specified.

Related articles

- USMT XML reference.

Feedback

Was this page helpful? Yes No

Provide product feedback

XML file requirements

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

When creating custom **.xml** files, note the following requirements:

- **The file must be in Unicode Transformation Format-8 (UTF-8).** Save the file in this format and the following syntax must be specified at the beginning of each **.xml** file:

  ```xml
  <?xml version="1.0" encoding="UTF-8"?>
  ```

- **The file must have a unique migration URL ID**. The URL ID of each file that is specified on the command line must be different. If two migration **.xml** files have the same URL ID, the second **.xml** file that is specified on the command line isn't processed. The second file isn't processed because USMT uses the URL ID to define the components within the file. For example, the following syntax must be specified at the beginning of each file:

  ```xml
  <?xml version="1.0" encoding="UTF-8"?>
  <migration
  urlid="http://www.microsoft.com/migration/1.0/migxmlext/<CustomFileName
  >">
  ```

- **Each component in the file must have a display name in order for it to appear in the Config.xml file.** This condition is because the `Config.xml` file defines the components by the display name and the migration URL ID. For example, specify the following syntax:

  ```xml
  <displayName>My Application</displayName>
  ```

For examples of custom **.xml** files, see Custom XML examples.

Feedback

Was this page helpful? Yes No

Provide product feedback

Recognized environment variables

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

When the XML files `MigDocs.xml`, `MigApp.xml`, and `MigUser.xml` are used, the environment variables can be used to identify folders that can be different on different computers. Constant special item ID list (CSIDL) values provide a way to identify folders that applications use frequently but could have different names or locations on any given computer. For example, the **Documents** folder could be `C:\Users\<Username>\Documents` on one computer and `C:\Users\<Username>\My Documents` on another. The asterisk (*) wildcard character can be used in the `MigUser.xml`, `MigApp.xml` and `MigDoc.xml` files. However, the asterisk (*) wildcard character can't be used in the `Config.xml` file.

Variables that are processed for the operating system and in the context of each user

These variables can be used within sections in the **.xml** files with `context=UserAndSystem`, `context=User`, and `context=System`.

Expand table

Variable	Explanation
ALLUSERSAPPDATA	Same as **CSIDL_COMMON_APPDATA**.
ALLUSERSPROFILE	Refers to `%PROFILESFOLDER%\Public` or `%PROFILESFOLDER%\all users`.
COMMONPROGRAMFILES	Same as **CSIDL_PROGRAM_FILES_COMMON**.
COMMONPROGRAMFILES(X86)	Refers to the `C:\Program Files (x86)\Common Files` folder on 64-bit systems.
CSIDL_COMMON_ADMINTOOLS	Version 10.0. The file-system directory that contains administrative tools for all users of the computer.
CSIDL_COMMON_ALTSTARTUP	The file-system directory that corresponds to the non-localized Startup program group for all users.
CSIDL_COMMON_APPDATA	The file-system directory that contains application data for all users. A typical path Windows is `C:\ProgramData`.
CSIDL_COMMON_DESKTOPDIRECTORY	The file-system directory that contains files and folders that appear on the desktop for all users. A typical path is

Variable	Explanation
	`C:\Users\Public\Desktop`.
CSIDL_COMMON_DOCUMENTS	The file-system directory that contains documents that are common to all users. A typical path is `C:\Users\Public\Documents`.
CSIDL_COMMON_FAVORITES	The file-system directory that serves as a common repository for favorites common to all users. A typical path is C:\Users\Public\Favorites.
CSIDL_COMMON_MUSIC	The file-system directory that serves as a repository for music files common to all users. A typical path is `C:\Users\Public\Music`.
CSIDL_COMMON_PICTURES	The file-system directory that serves as a repository for image files common to all users. A typical path is `C:\Users\Public\Pictures`.
CSIDL_COMMON_PROGRAMS	The file-system directory that contains the directories for the common program groups that appear on the **Start** menu for all users. A typical path is `C:\ProgramData\Microsoft\Windows\Start Menu\Programs`.
CSIDL_COMMON_STARTMENU	The file-system directory that contains the programs and folders that appear on the **Start** menu for all users. A typical path in Windows is `C:\ProgramData\Microsoft\Windows\Start Menu`.
CSIDL_COMMON_STARTUP	The file-system directory that contains the programs that appear in the Startup folder for all users. A typical path is `C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup`.
CSIDL_COMMON_TEMPLATES	The file-system directory that contains the templates that are available to all users. A typical path is `C:\ProgramData\Microsoft\Windows\Templates`.
CSIDL_COMMON_VIDEO	The file-system directory that serves as a repository for video files common to all users. A typical path is `C:\Users\Public\Videos`.
CSIDL_DEFAULT_APPDATA	Refers to the `Appdata` folder inside `%DEFAULTUSERPROFILE%`.
CSIDL_DEFAULT_LOCAL_APPDATA	Refers to the local `Appdata` folder inside `%DEFAULTUSERPROFILE%`.
CSIDL_DEFAULT_COOKIES	Refers to the Cookies folder inside `%DEFAULTUSERPROFILE%`.

Variable	Explanation
CSIDL_DEFAULT_CONTACTS	Refers to the Contacts folder inside `%DEFAULTUSERPROFILE%`.
CSIDL_DEFAULT_DESKTOP	Refers to the Desktop folder inside `%DEFAULTUSERPROFILE%`.
CSIDL_DEFAULT_DOWNLOADS	Refers to the Downloads folder inside `%DEFAULTUSERPROFILE%`.
CSIDL_DEFAULT_FAVORITES	Refers to the Favorites folder inside `%DEFAULTUSERPROFILE%`.
CSIDL_DEFAULT_HISTORY	Refers to the History folder inside `%DEFAULTUSERPROFILE%`.
CSIDL_DEFAULT_INTERNET_CACHE	Refers to the Internet Cache folder inside `%DEFAULTUSERPROFILE%`.
CSIDL_DEFAULT_PERSONAL	Refers to the Personal folder inside `%DEFAULTUSERPROFILE%`.
CSIDL_DEFAULT_MYDOCUMENTS	Refers to the Documents folder inside `%DEFAULTUSERPROFILE%`.
CSIDL_DEFAULT_MYPICTURES	Refers to the Pictures folder inside `%DEFAULTUSERPROFILE%`.
CSIDL_DEFAULT_MYMUSIC	Refers to the Music folder inside `%DEFAULTUSERPROFILE%`.
CSIDL_DEFAULT_MYVIDEO	Refers to the Videos folder inside `%DEFAULTUSERPROFILE%`.
CSIDL_DEFAULT_RECENT	Refers to the Recent folder inside `%DEFAULTUSERPROFILE%`.
CSIDL_DEFAULT_SENDTO	Refers to the Send To folder inside `%DEFAULTUSERPROFILE%`.
CSIDL_DEFAULT_STARTMENU	Refers to the Start Menu folder inside `%DEFAULTUSERPROFILE%`.
CSIDL_DEFAULT_PROGRAMS	Refers to the Programs folder inside `%DEFAULTUSERPROFILE%`.
CSIDL_DEFAULT_STARTUP	Refers to the Startup folder inside `%DEFAULTUSERPROFILE%`.
CSIDL_DEFAULT_TEMPLATES	Refers to the Templates folder inside `%DEFAULTUSERPROFILE%`.
CSIDL_DEFAULT_QUICKLAUNCH	Refers to the Quick Launch folder inside `%DEFAULTUSERPROFILE%`.

Variable	Explanation
CSIDL_FONTS	A virtual folder containing fonts. A typical path is `C:\Windows\Fonts`.
CSIDL_PROGRAM_FILESX86	The Program Files folder on 64-bit systems. A typical path is `C:\Program Files (x86)`.
CSIDL_PROGRAM_FILES_COMMONX86	A folder for components that are shared across applications on 64-bit systems. A typical path is `C:\Program Files (x86)\Common`.
CSIDL_PROGRAM_FILES	The Program Files folder. A typical path is `C:\Program Files`.
CSIDL_PROGRAM_FILES_COMMON	A folder for components that are shared across applications. A typical path is `C:\Program Files\Common`.
CSIDL_RESOURCES	The file-system directory that contains resource data. A typical path is `C:\Windows\Resources`.
CSIDL_SYSTEM	The Windows System folder. A typical path is `C:\Windows\System32`.
CSIDL_WINDOWS	The Windows directory or system root path. This value corresponds to the `%WINDIR%` or `%SYSTEMROOT%` environment variables. A typical path is `C:\Windows`.
DEFAULTUSERPROFILE	Refers to the value in `HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\ProfileList [DefaultUserProfile]`.
PROFILESFOLDER	Refers to the value in `HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\ProfileList [ProfilesDirectory]`.
PROGRAMFILES	Same as **CSIDL_PROGRAM_FILES**.
PROGRAMFILES(X86)	Refers to the `C:\Program Files (x86)` folder on 64-bit systems.
SYSTEM	Refers to `%WINDIR%\system32`.
SYSTEM16	Refers to `%WINDIR%\system`.
SYSTEM32	Refers to `%WINDIR%\system32`.
SYSTEMDRIVE	The drive that holds the Windows folder. This value is a drive name and not a folder name (`C:` not `C:\`).
SYSTEMPROFILE	Refers to the value in `HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\ProfileList\S-1-5-18 [ProfileImagePath]`.

Variable	Explanation
SYSTEMROOT	Same as **WINDIR**.
WINDIR	Refers to the Windows folder located on the system drive.

Variables that are recognized only in the user context

These variables can be used in the **.xml** files within sections with `context=User` and `context=UserAndSystem`.

Expand table

Variable	Explanation
APPDATA	Same as **CSIDL_APPDATA**.
CSIDL_ADMINTOOLS	The file-system directory that is used to store administrative tools for an individual user. The Microsoft Management Console (MMC) saves customized consoles to this directory, which roams with the user profile.
CSIDL_ALTSTARTUP	The file-system directory that corresponds to the user's non-localized Startup program group.
CSIDL_APPDATA	The file-system directory that serves as a common repository for application-specific data. A typical path is `C:\Users\<username>\AppData\Roaming`.
CSIDL_BITBUCKET	The virtual folder that contains the objects in the user's Recycle Bin.
CSIDL_CDBURN_AREA	The file-system directory acting as a staging area for files waiting to be written to CD. A typical path is `C:\Users\<username>\AppData\Local\Microsoft\Windows\MasteredBurning\DiscBurning`.
CSIDL_CONNECTIONS	The virtual folder representing Network Connections that contains network and dial-up connections.
CSIDL_CONTACTS	This value refers to the Contacts folder in **%CSIDL_PROFILE%**.
CSIDL_CONTROLS	The virtual folder that contains icons for the Control Panel items.
CSIDL_COOKIES	The file-system directory that serves as a common repository for Internet cookies. A typical path is `C:\Users\`

Variable	Explanation
	`<username>\AppData\Roaming\Microsoft\Windows\Cookies`.
CSIDL_DESKTOP	The virtual folder representing the Windows desktop.
CSIDL_DESKTOPDIRECTORY	The file-system directory used to physically store file objects on the desktop, which shouldn't be confused with the desktop folder itself. A typical path is `C:\Users\<username>\Desktop`.
CSIDL_DRIVES	The virtual folder representing **This PC** that contains everything on the local computer: storage devices, printers, and Control Panel. The folder could also contain mapped network drives.
CSIDL_FAVORITES	The file-system directory that serves as a common repository for the user's favorites. A typical path is `C:\Users\<username>\Favorites`.
CSIDL_HISTORY	The file-system directory that serves as a common repository for Internet history items.
CSIDL_INTERNET	A virtual folder for Internet Explorer.
CSIDL_INTERNET_CACHE	The file-system directory that serves as a common repository for temporary Internet files. A typical path is `C:\Users\<username>\AppData\Local\Microsoft\Windows\Temporary Internet Files`
CSIDL_LOCAL_APPDATA	The file-system directory that serves as a data repository for local, non-roaming applications. A typical path is `C:\Users\<username>\AppData\Local`.
CSIDL_MYDOCUMENTS	The virtual folder representing the **Documents** folder.A typical path is `C:\Users\<username>\Documents`.
CSIDL_MYMUSIC	The file-system directory that serves as a common repository for music files. A typical path is `C:\Users\<username>\Music`.
CSIDL_MYPICTURES	The file-system directory that serves as a common repository for image files. A typical path is `C:\Users\<username>\Pictures`.
CSIDL_MYVIDEO	The file-system directory that serves as a common repository for video files. A typical path is `C:\Users\<username>\Videos`.
CSIDL_NETHOOD	A file-system directory that contains the link objects that could exist in the **Network** virtual folder. It isn't the same as *CSIDL_NETWORK*, which represents the network namespace root. A typical path is `C:\Users\<username>\AppData\Roaming\Microsoft\Windows\Network Shortcuts`.
CSIDL_NETWORK	A virtual folder representing the **Network** desktop item, the root of

Variable	Explanation
	the network namespace hierarchy.
CSIDL_PERSONAL	The virtual folder representing the **<User>** desktop item. This value is equivalent to **CSIDL_MYDOCUMENTS**. A typical path is `C:\User\<username>\Documents`.
CSIDL_PLAYLISTS	The virtual folder used to store play albums, typically `C:\Users\<username>\Music\Playlists`.
CSIDL_PRINTERS	The virtual folder that contains installed printers.
CSIDL_PRINTHOOD	The file-system directory that contains the link objects that can exist in the Printers virtual folder. A typical path is `C:\Users\<username>\AppData\Roaming\Microsoft\Windows\Printer Shortcuts`.
CSIDL_PROFILE	The user's profile folder. A typical path is `C:\Users\<username>`.
CSIDL_PROGRAMS	The file-system directory that contains the user's program groups, which are themselves file-system directories. A typical path is `C:\Users\<username>\AppData\Roaming\Microsoft\Windows\Start Menu\Programs`.
CSIDL_RECENT	The file-system directory that contains shortcuts to the user's most recently used documents. A typical path is `C:\Users\<username>\AppData\Roaming\Microsoft\Windows\Recent`.
CSIDL_SENDTO	The file-system directory that contains **Send To** menu items. A typical path is `C:\Users\<username>\AppData\Roaming\Microsoft\Windows\SendTo`.
CSIDL_STARTMENU	The file-system directory that contains **Start** menu items. A typical path is `C:\Users\<username>\AppData\Roaming\Microsoft\Windows\Start Menu`.
CSIDL_STARTUP	The file-system directory that corresponds to the user's Startup program group. A typical path is `C:\Users\<username>\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup`.
CSIDL_TEMPLATES	The file-system directory that serves as a common repository for document templates. A typical path is `C:\Users\<username>\AppData\Roaming\Microsoft\Windows\Templates`.
HOMEPATH	Same as the standard environment variable.
TEMP	The temporary folder on the computer. A typical path is `%USERPROFILE%\AppData\Local\Temp`.

Variable	Explanation
TMP	The temporary folder on the computer. A typical path is `%USERPROFILE%\AppData\Local\Temp`.
USERPROFILE	Same as **CSIDL_PROFILE**.
USERSID	Represents the current user-account security identifier (SID). For example, `S-1-5-21-1714567821-1326601894-715345443-1026`.

Related articles

USMT XML reference

Feedback

Was this page helpful? Yes No

Provide product feedback

XML elements library

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

This article describes the XML elements and helper functions that can be employed to author migration **.xml** files to use with User State Migration Tool (USMT). This article assumes a basic knowledge of XML.

In addition to XML elements and helper functions, this article:

- Describes how to specify encoded locations and locations patterns.
- Functions that are for internal USMT use only.
- The version tags that can be used with helper functions.

Elements and helper functions

The following table describes the XML elements and helper functions can be used with USMT.

Expand table

Elements A-K	Elements L-Z	Helper functions
<addObjects>	<library>	<condition> functions
<attributes>	<location>	<content> functions
<bytes>	<locationModify>	<contentModify> functions
<commandLine>	<_locDefinition>	<include> and <exclude> filter functions
<component>	<manufacturer>	<locationModify> functions
<condition>	<merge>	<merge> functions
<conditions>	<migration>	<script> functions
<content>	<namedElements>	Internal USMT functions
<contentModify>	<object>	
<description>	<objectSet>	
<destinationCleanup>	<path>	
<detect>	<paths>	
<detects>	<pattern>	
<detection>	<processing>	
<displayName>	<plugin>	
<environment>	<role>	
<exclude>	<rules>	
<excludeAttributes>	<script>	
<extensions>	<text>	
<extension>	<unconditionalExclude>	
<externalProcess>	<variable>	
<icon>	<version>	
<include>	<windowsObjects>	
<includeAttribute>		

<addObjects>

The **<addObjects>** element emulates the existence of one or more objects on the source computer. The child **<object>** elements provide the details of the emulated objects. If the content is a **<script>** element, the result of the invocation is an array of objects.

- **Number of occurrences:** unlimited
- **Parent elements:** <rules>
- **Required child elements:** <object> In addition, <location> and <attribute> must be specified as child elements of this **<object>** element.
- **Optional child elements:** <conditions>, <condition>, <script>

Syntax:

```
XML
```

```
<addObjects>
</addObjects>
```

The following example is from the `MigApp.xml` file:

XML

```
<addObjects>
   <object>
      <location
type="Registry">%HklmWowSoftware%\Microsoft\Office\16.0\Common\Migration\Off
ice [UpgradeVersion]</location>
      <attributes>DWORD</attributes>
      <bytes>0B000000</bytes>
   </object>
   <object>
      <location
type="Registry">%HklmWowSoftware%\Microsoft\Office\16.0\Common\Migration\Off
ice [Lang]</location>
      <attributes>DWORD</attributes>
      <bytes>00000000</bytes>
   </object>
</addObjects>
```

<attributes>

The **<attributes>** element defines the attributes for a registry key or file.

- **Number of occurrences:** once for each <object>
- **Parent elements:** <object>
- **Child elements:** none

Syntax:

XML

```
<attributes>Content</attributes>
```

Expand table

Setting	Required?	Value
Content	Yes	The content depends on the type of object specified.

Setting	Required?	Value

		• For files, the content can be a string containing any of the following attributes separated by commas:
		  ◦ Archive
		  ◦ Read-only
		  ◦ System
		  ◦ Hidden
		• For registry keys, the content can be one of the following types:
		  ◦ None
		  ◦ String
		  ◦ ExpandString
		  ◦ Binary
		  ◦ Dword
		  ◦ REG_SZ

The following example is from the `MigApp.xml` file:

XML

```xml
<object>
   <location
type="Registry">%HklmWowSoftware%\Microsoft\Office\16.0\Common\Migration\Off
ice [Lang]</location>
   <attributes>DWORD</attributes>
   <bytes>00000000</bytes>
</object>
```

<bytes>

The **<bytes>** element can only be specified for files because, if **<location>** corresponds to a registry key or a directory, then **<bytes>** is ignored.

- **Number of occurrences:** zero or one
- **Parent elements:** <object>
- **Child elements:** none

Syntax:

XML

```xml
<bytes string="Yes|No" expand="Yes|No">Content</bytes>
```

Expand table

Setting	Required?	Value
string	No, default is No	Determines whether *Content* should be interpreted as a string or as bytes.
expand	No (default = Yes	When the expand parameter is **Yes**, the content of the **<bytes>** element is first expanded in the context of the source computer and then interpreted.
Content	Yes	Depends on the value of the string. • When the string is **Yes**: the content of the **<bytes>** element is interpreted as a string. • When the string is **No**: the content of the **<bytes>** element is interpreted as bytes. Every two characters represent the hexadecimal value of a byte. For example, `616263` is the representation for the `abc` ANSI string. A complete representation of the UNICODE string `abc` including the string terminator would be: `6100620063000000`.

The following example is from the `MigApp.xml` file:

XML

```xml
<object>
   <location
type="Registry">%HklmWowSoftware%\Microsoft\Office\16.0\Common\Migration\Off
ice [Lang]</location>
   <attributes>DWORD</attributes>
   <bytes>00000000</bytes>
</object>
```

<commandLine>

The **<commandLine>** element might be used to start or stop a service or application before or after running the **ScanState** and **LoadState** tools.

- **Number of occurrences:** unlimited
- **Parent elements:** <externalProcess>
- **Child elements:** none

Syntax:

```XML
<commandLine>CommandLineString</commandLine>
```

Expand table

Setting	Required?	Value
CommandLineString	Yes	A valid command line.

<component>

The **<component>** element is required in a custom **.xml** file. This element defines the most basic construct of a migration **.xml** file. For example, in the `MigApp.xml` file, **Microsoft Office 2016** is a component that contains another component, **Microsoft Office Access 2016**. The child elements can be used to define the component.

A component can be nested inside another component; that is, the **<component>** element can be a child of the **<role>** element within the **<component>** element in two cases:

1. When the parent **<component>** element is a container
2. If the child **<component>** element has the same role as the parent **<component>** element.

- **Number of occurrences:** Unlimited
- **Parent elements:** <migration>, <role>
- **Required child elements:** <role>, <displayName>
- **Optional child elements:** <manufacturer>, <version>, <description>, <paths>, <icon>, <environment>, <extensions>

Syntax:

```XML
<component type="System|Application|Device|Documents"
context="User|System|UserAndSystem" defaultSupported="TRUE|FALSE|YES|NO"
hidden="Yes|No">
</component>
```

Expand table

Setting	Required?	Value
type	Yes	The following items can be used to group settings, and define the type of the component. • **System:** Operating system settings. All Windows components are defined by this type. When **type="System"** and **defaultSupported="FALSE"**, the settings don't migrate unless there's an equivalent component in the **.xml** files that is specified on the `LoadState.exe` command line. For example, the default `MigSys.xml` file contains components with **type="System"** and **defaultSupported="FALSE"**. If this file is specified on the `ScanState.exe` command line, the file must also be specified on the `LoadState.exe` command line for the settings to migrate. The file must be specified because the `LoadState.exe` tool must detect an equivalent component. That is, the component must have the same migration urlid of the **.xml** file and an identical display name. Otherwise, the **LoadState** tool doesn't migrate those settings from the store. This setting is helpful because a store can be used for destination computers that are the same or different version of Windows as the source computer. • **Application:** Settings for an application. • **Device:** Settings for a device. • **Documents:** Specifies files.
context	No Default = UserAndSystem	Defines the scope of this parameter; that is, whether to process this component in the context of the specific user, across the entire operating system, or both. The largest possible scope is set by the **<component>** element. For example, if a **<component>** element has a context of **User** and a **<rules>** element had a context of **UserAndSystem**, then the **<rules>** element would act as though it has a context of **User**. If a **<rules>** element has a context of **System**, it would act as though the **<rules>** element isn't there. • **User**: Evaluates the component for each user. • **System**: Evaluates the component only once for the system. • **UserAndSystem**: Evaluates the component for the entire operating system and each user.

Setting	Required?	Value
defaultSupported	No (default = TRUE)	Can be any of **TRUE**, **FALSE**, **YES**, or **NO**. If this parameter is **FALSE** (or **NO**), the component isn't migrated unless there's an equivalent component on the destination computer. When **type="System"** and **defaultSupported="FALSE"**, the settings aren't migrated unless there's an equivalent component in the **.xml** files that are specified on the `LoadState.exe` command line. For example, the default `MigSys.xml` file contains components with **type="System"** and **defaultSupported="FALSE"**. If this file is specified on the `ScanState.exe` command line, the file must also be specified on the `LoadState.exe` command line for the settings to migrate. The file has to be specified in both command lines because the **LoadState** tool must detect an equivalent component. That is, the component must have the same migration urlid of the **.xml** file and an identical display name or the **LoadState** tool doesn't migrate those settings from the store. This setting is helpful because a store can be used for destination computers that are the same or different version of Windows as the source computer.
hidden		This parameter is for internal USMT use only.

For an example, see any of the default migration **.xml** files.

<condition>

Although the **<condition>** element under the **<detect>**, **<objectSet>**, and **<addObjects>** elements is still supported, Microsoft recommends to no longer use the **<condition>** element because it might be deprecated in future versions of USMT. If the **<condition>** element is deprecated, it would require a rewrite of any scripts that use the **<condition>** element. Instead, if a condition needs to be used within the **<objectSet>** and **<addObjects>** elements, Microsoft recommends using the more powerful <conditions> element. The **<conditions>** element allows for formulation of complex Boolean statements.

The **<condition>** element has a Boolean result. This element can be used to specify the conditions in which the parent element is evaluated. If any of the present conditions return **FALSE**, the parent element isn't be evaluated.

- **Number of occurrences:** unlimited.
- **Parent elements:** <conditions>, <detect>, <objectSet>, <addObjects>

- **Child elements:** none
- **Helper functions:** The following <condition> functions can be used with this element: `DoesOSMatch`, `IsNative64Bit()`, `IsOSLaterThan`, `IsOSEarlierThan`, `DoesObjectExist`, `DoesFileVersionMatch`, `IsFileVersionAbove`, `IsFileVersionBelow`, `IsSystemContext`, `DoesStringContentEqual`, `DoesStringContentContain`, `IsSameObject`, `IsSameContent`, and `IsSameStringContent`.

Syntax:

XML

```
<condition negation="Yes|No">ScriptName</condition>
```

Expand table

Setting	Required?	Value
negation	No Default = No	"**Yes**" reverses the True/False value of the condition.
ScriptName	Yes	A script that is defined within this migration section.

For example, in the following code sample, the **<condition>** elements, **A** and **B**, are joined together by the **AND** operator because they are in separate **<conditions>** sections:

XML

```
<detection>
   <conditions>
      <condition>A</condition>
   </conditions>
   <conditions operation="AND">
      <condition>B</condition>
   </conditions>
</detection>
```

However, in the following code sample, the **<condition>** elements, **A** and **B**, are joined together by the **OR** operator because they are in the same **<conditions>** section.

XML

```
<detection>
   <conditions>
      <condition>A</condition>
```

```
      <condition>B</condition>
    </conditions>
  </detection>
```

<condition> functions

The <**condition**> functions return a Boolean value. These elements can be used in <**addObjects**> conditions.

- Operating system version functions
- Object content functions

Operating system version functions

- **DoesOSMatch**

  All matches are case insensitive.

  Syntax: `DoesOSMatch("OSType","OSVersion")`

  Expand table

Setting	Required?	Value
OSType	Yes	The only valid value for this setting is **NT**. However, this setting must be set for the <**condition**> functions to work correctly.
OSVersion	Yes	The major version, minor version, build number and corrected service diskette version separated by periods. For example, `5.0.2600.Service Pack 1`. Partial specification of the version can also be specified with a pattern such as `5.0.*`.

  For example:

```
XML
<condition>MigXmlHelper.DoesOSMatch("NT","\*")</condition>
```

- **IsNative64Bit**

  The **IsNative64Bit** function returns **TRUE** if the migration process is running as a native 64-bit process; that is, a process running on a 64-bit system without Windows on Windows (WOW). Otherwise, it returns **FALSE**.

- **IsOSLaterThan**

  All comparisons are case insensitive.

  Syntax: `IsOSLaterThan("OSType","OSVersion")`

  Expand table

Setting	Required?	Value
OSType	Yes	Can be **9x** or **NT**. If *OSType* doesn't match the type of the current operating system, then it returns **FALSE**. For example, if the current operating system is Windows NT-based and *OSType* is "**9x**", the result is **FALSE**.
OSVersion	Yes	The major version, minor version, build number, and corrected service diskette version separated by periods. For example, `5.0.2600.Service Pack 1`. Partial specification of the version can also be specified but no pattern is allowed such as `5.0`. The **IsOSLaterThan** function returns **TRUE** if the current operating system is later than or equal to *OSVersion*.

  For example:

```xml
<condition negation="Yes">MigXmlHelper.IsOSLaterThan("NT","6.0")
</condition>
```

- **IsOSEarlierThan**

  All comparisons are case insensitive.

  Syntax: `IsOSEarlierThan("OSType","OSVersion")`

  Expand table

Setting	Required?	Value
OSType	Yes	Can be **9x** or **NT**. If *OSType* doesn't match the type of the current operating system, then it returns **FALSE**. For example, if the current operating system is Windows NT-based and *OSType* is "**9x**" the result is **FALSE**.
OSVersion	Yes	The major version, minor version, build number, and corrected service diskette version separated by periods. For example, `5.0.2600.Service Pack 1`. Partial specification of the version can

Setting	Required?	Value
		also be specified but no pattern is allowed such as `5.0`. The **IsOSEarlierThan** function returns **TRUE** if the current operating system is earlier than *OSVersion*.

Object content functions

- **DoesObjectExist**

  The DoesObjectExist function returns **TRUE** if any object exists that matches the location pattern. Otherwise, it returns **FALSE**. The location pattern is expanded before attempting the enumeration.

  Syntax: `DoesObjectExist("ObjectType","EncodedLocationPattern")`

  Expand table

Setting	Required?	Value
ObjectType	Yes	Defines the object type. Can be File or Registry.
EncodedLocationPattern	Yes	The location pattern. Environment variables are allowed.

  For an example of this element, see the `MigApp.xml` file.

- **DoesFileVersionMatch**

  The pattern check is case insensitive.

  Syntax:
  `DoesFileVersionMatch("EncodedFileLocation","VersionTag","VersionValue")`

  Expand table

Setting	Required?	Value
EncodedFileLocation	Yes	The location pattern for the file that is checked. Environment variables are allowed.
VersionTag	Yes	The version tag value that is checked.
VersionValue	Yes	A string pattern. For example, "Microsoft*".

  For example:

```xml
<condition>MigXmlHelper.DoesFileVersionMatch("%MSNMessengerInstPath%\\m
snmsgr.exe","ProductVersion","6.\*")</condition>
<condition>MigXmlHelper.DoesFileVersionMatch("%MSNMessengerInstPath%\\m
snmsgr.exe","ProductVersion","7.\*")</condition>
```

- **IsFileVersionAbove**

  The **IsFileVersionAbove** function returns **TRUE** if the version of the file is higher than *VersionValue*.

  Syntax: `IsFileVersionAbove("EncodedFileLocation","VersionTag","VersionValue")`

  Expand table

Setting	Required?	Value
EncodedFileLocation	Yes	The location pattern for the file that is checked. Environment variables are allowed.
VersionTag	Yes	The version tag value that is checked.
VersionValue	Yes	The value to compare to. A pattern can't be specified.

- **IsFileVersionBelow**

  Syntax: `IsFileVersionBelow("EncodedFileLocation","VersionTag","VersionValue")`

  Expand table

Setting	Required?	Value
EncodedFileLocation	Yes	The location pattern for the file that is checked. Environment variables are allowed.
VersionTag	Yes	The version tag value that is checked.
VersionValue	Yes	The value to compare to. A pattern can't be specified.

- **IsSystemContext**

  The **IsSystemContext** function returns **TRUE** if the current context is "**System**". Otherwise, it returns **FALSE**.

  Syntax: `IsSystemContext()`

- **DoesStringContentEqual**

The **DoesStringContentEqual** function returns **TRUE** if the string representation of the given object is identical to `StringContent`.

Syntax: `DoesStringContentEqual("ObjectType","EncodedLocation","StringContent")`

Expand table

Setting	Required?	Value
ObjectType	Yes	Defines the type of object. Can be File or Registry.
EncodedLocationPattern	Yes	The encoded location for the object that is examined. Environment variables can be specified.
StringContent	Yes	The string that is checked against.

For example:

```
XML

<condition
negation="Yes">MigXmlHelper.DoesStringContentEqual("File","%USERNAME%",
"")</condition>
```

- **DoesStringContentContain**

The **DoesStringContentContain** function returns **TRUE** if there is at least one occurrence of *StrToFind* in the string representation of the object.

Syntax: `DoesStringContentContain("ObjectType","EncodedLocation","StrToFind")`

Expand table

Setting	Required?	Value
ObjectType	Yes	Defines the type of object. Can be File or Registry.
EncodedLocationPattern	Yes	The encoded location for the object that is examined. Environment variables can be specified.
StrToFind	Yes	A string that is searched inside the content of the given object.

- **IsSameObject**

The **IsSameObject** function returns **TRUE** if the given encoded locations resolve to the same physical object. Otherwise, it returns **FALSE**.

Syntax: `IsSameObject("ObjectType","EncodedLocation1","EncodedLocation2")`

Expand table

Setting	Required?	Value
ObjectType	Yes	Defines the type of object. Can be File or Registry.
EncodedLocation1	Yes	The **encoded location** for the first object. Environment variables can be specified.
EncodedLocation2	Yes	The **encoded location** for the second object. Environment variables can be specified.

For example:

```XML
<objectSet>
   <condition negation="Yes">MigXmlHelper.IsSameObject("File","%CSIDL_FAVORITES%","%CSIDL_COMMON_FAVORITES%")</condition>
   <pattern type="File">%CSIDL_FAVORITES%\* [*]</pattern>
</objectSet>
```

- **IsSameContent**

The **IsSameContent** function returns **TRUE** if the given objects have the same content. Otherwise, it returns **FALSE**. The content is compared byte by byte.

Syntax:
`IsSameContent("ObjectType1","EncodedLocation1","ObjectType2","EncodedLocation2")`

Expand table

Setting	Required?	Value
ObjectType1	Yes	Defines the type of the first object. Can be File or Registry.
EncodedLocation1	Yes	The **encoded location** for the first object. Environment variables can be specified.
ObjectType2	Yes	Defines the type of the second object. Can be File or Registry.
EncodedLocation2	Yes	The **encoded location** for the second object. Environment variables can be specified.

- **IsSameStringContent**

  The **IsSameStringContent** function returns **TRUE** if the given objects have the same content. Otherwise, it returns **FALSE**. The content is interpreted as a string.

  Syntax:
  `IsSameStringContent("ObjectType1","EncodedLocation1","ObjectType2","EncodedLocation2")`

Expand table

Setting	Required?	Value
ObjectType1	Yes	Defines the type of the first object. Can be File or Registry.
EncodedLocation1	Yes	The **encoded location** for the first object. Environment variables can be specified.
ObjectType2	Yes	Defines the type of the second object. Can be File or Registry.
EncodedLocation2	Yes	The **encoded location** for the second object. Environment variables can be specified.

<conditions>

The **<conditions>** element returns a Boolean result that is used to specify the conditions in which the parent element is evaluated. USMT evaluates the child elements, and then joins their results using the operators **AND** or **OR** according to the operation parameter.

- **Number of occurrences:** Unlimited inside another **<conditions>** element. Limited to one occurrence in <detection>, <rules>, <addObjects>, and <objectSet>
- **Parent elements:** <conditions>, <detection>, <environment>, <rules>, <addObjects>, and <objectSet>
- **Child elements:** <conditions>, <condition>

Syntax:

```XML
<conditions operation="AND|OR">
</conditions>
```

Expand table

Setting	Required?	Value
operation	No, default = AND	Defines the Boolean operation that is performed on the results that are obtained from the child elements.

The following example is from the `MigApp.xml` file:

```XML
<environment name="GlobalEnv">
   <conditions>
      <condition negation="Yes">MigXmlHelper.IsNative64Bit()</condition>
   </conditions>
   <variable name="HklmWowSoftware">
   <text>HKLM\Software</text>
   </variable>
</environment>
```

<content>

The **<content>** element can be used to specify a list of object patterns to obtain an object set from the source computer. Each **<objectSet>** within a **<content>** element is evaluated. For each resulting object pattern list, the objects that match it are enumerated and their content is filtered by the filter parameter. The resulting string array is the output for the **<content>** element. The filter script returns an array of locations. The parent **<objectSet>** element can contain multiple child **<content>** elements.

- **Number of occurrences:** unlimited
- **Parent elements:** <objectSet>
- **Child elements:** <objectSet>
- **Helper functions:** The following <content> functions can be used with this element: `ExtractSingleFile`, `ExtractMultipleFiles`, and `ExtractDirectory`.

Syntax:

```XML
<content filter="ScriptInvocation">
</content>
```

Expand table

Setting	Required?	Value
filter	Yes	A script followed by any number of string arguments that are separated by a comma and enclosed in parenthesis. For example, `MyScripts.AScript ("Arg1","Arg2")`. The script is called for each object that is enumerated by the object sets in the **<include>** rule. The filter script returns a Boolean value. If the return value is **TRUE**, the object is migrated. If it's **FALSE**, it isn't migrated.

<content> functions

The following functions generate patterns out of the content of an object. These functions are called for every object that the parent **<ObjectSet>** element is enumerating.

- **ExtractSingleFile**

  If the registry value is a **MULTI-SZ**, only the first segment is processed. The returned pattern is the encoded location for a file that must exist on the system. If the specification is correct in the registry value, but the file doesn't exist, this function returns **NULL**.

  Syntax: `ExtractSingleFile(Separators,PathHints)`

  Expand table

Setting	Required?	Value
Separators	Yes	A list of possible separators that might follow the file specification in this registry value name. For example, if the content is **"C:\Windows\Notepad.exe,-2"**, the separator is a comma. **NULL** can be specified.
PathHints	Yes	A list of extra paths, separated by colons (`;`), where the function looks for a file matching the current content. For example, if the content is **"Notepad.exe"** and the path is the **%Path%** environment variable, the function finds **Notepad.exe** in `%windir%` and returns **"c:\Windows [Notepad.exe]"**. **NULL** can be specified.

  For example:

  XML

```
<content filter="MigXmlHelper.ExtractSingleFile(',','%system%')">
```

and

```XML
<content
filter="MigXmlHelper.ExtractSingleFile(NULL,'%CSIDL_COMMON_FONTS%')">
```

- **ExtractMultipleFiles**

  The **ExtractMultipleFiles** function returns multiple patterns, one for each file that is found in the content of the given registry value. If the registry value is a **MULTI-SZ**, the **MULTI-SZ** separator is considered a separator by default. therefore, for **MULTI-SZ**, the **<Separators>** argument must be **NULL**.

  The returned patterns are the encoded locations for files that must exist on the source computer. If the specification is correct in the registry value but the file doesn't exist, it isn't included in the resulting list.

  Syntax: `ExtractMultipleFiles(Separators,PathHints)`

  Expand table

Setting	Required?	Value
Separators	Yes	A list of possible separators that might follow the file specification in this registry value name. For example, if the content is **"C:\Windows\Notepad.exe,-2"**, the separator is a comma. This parameter must be NULL when processing **MULTI-SZ** registry values.
PathHints	Yes	A list of extra paths, separated by colons (`;`), where the function looks for a file matching the current content. For example, if the content is **"Notepad.exe"** and the path is the **%Path%** environment variable, the function finds **Notepad.exe** in `%windir%` and returns **"c:\Windows [Notepad.exe]"**. **NULL** can be specified.

- **ExtractDirectory**

  The **ExtractDirectory** function returns a pattern that is the encoded location for a directory that must exist on the source computer. If the specification is correct in the registry value, but the directory doesn't exist, this function returns **NULL**. If it's processing a registry value that is a **MULTI-SZ**, only the first segment is processed.

Syntax: `ExtractDirectory(Separators,LevelsToTrim,PatternSuffix)`

Expand table

Setting	Required?	Value
Separators	No	A list of possible separators that might follow the file specification in this registry value name. For example, if the content is **"C:\Windows\Notepad.exe,-2"**, the separator is a comma. **NULL** must be specified when processing **MULTI-SZ** registry values.
LevelsToTrim	Yes	The number of levels to delete from the end of the directory specification. Use this function to extract a root directory when there's a registry value that points inside that root directory in a known location.
PatternSuffix	Yes	The pattern to add to the directory specification. For example, `* [*]`.

For example:

XML

```
<objectSet>
   <content filter='MigXmlHelper.ExtractDirectory (NULL, "1")'>
        <objectSet>
             <pattern
type="Registry">%HklmWowSoftware%\Classes\Software\RealNetworks\Prefere
nces\DT_Common []</pattern>
        </objectSet>
   </content>
</objectSet>
```

<contentModify>

The **<contentModify>** element modifies the content of an object before the object is written to the destination computer. For each **<contentModify>** element, there can be multiple **<objectSet>** elements. This element returns the new content of the object that is being processed.

- **Number of occurrences:** Unlimited
- **Parent elements:** <rules>
- **Required child elements:** <objectSet>

- **Helper functions**: The following <contentModify> functions can be used with this element: **ConvertToDWORD**, **ConvertToString**, **ConvertToBinary**, **KeepExisting**, **OffsetValue**, **SetValueByTable**, **MergeMultiSzContent**, and **MergeDelimitedContent**.

Syntax:

```XML
<contentModify script="ScriptInvocation">
</contentModify>
```

Expand table

Setting	Required?	Value
script	Yes	A script followed by any number of string arguments that are separated by a comma and enclosed in parenthesis. For example, `MyScripts.AScript ("Arg1","Arg2").` The script is called for each object that is enumerated by the object sets in the include rule. The filter script returns a Boolean value. If the return value is **TRUE**, the object is migrated. If it's **FALSE**, it isn't migrated.

<contentModify> functions

The following functions change the content of objects as they're migrated. These functions are called for every object that the parent **<ObjectSet>** element is enumerating.

- **ConvertToDWORD**

  The **ConvertToDWORD** function converts the content of registry values that are enumerated by the parent **<ObjectSet>** element to a DWORD. For example, **ConvertToDWORD** converts the string `"1"` to the DWORD `0x00000001`. If the conversion fails, then the value of **DefaultValueOnError** is applied.

  Syntax: `ConvertToDWORD(DefaultValueOnError)`

Expand table

Setting	Required?	Value
DefaultValueOnError	No	The value that is written into the value name if the conversion fails. **NULL** can be specified, and `0` is

Setting	Required?	Value
		written if the conversion fails.

- **ConvertToString**

  The **ConvertToString** function converts the content of registry values that match the parent **<ObjectSet>** element to a string. For example, it converts the DWORD `0x00000001` to the string **"1"**. If the conversion fails, then the value of **DefaultValueOnError** is applied.

  Syntax: `ConvertToString(DefaultValueOnError)`

  Expand table

Setting	Required?	Value
DefaultValueOnError	No	The value that is written into the value name if the conversion fails. **NULL** can be specified, and `0` is written if the conversion fails.

  For example:

```xml
<contentModify script="MigXmlHelper.ConvertToString('1')">
   <objectSet>
        <pattern type="Registry">HKCU\Control Panel\Desktop
[ScreenSaveUsePassword]</pattern>
   </objectSet>
</contentModify>
```

- **ConvertToBinary**

  The **ConvertToBinary** function converts the content of registry values that match the parent **<ObjectSet>** element to a binary type.

  Syntax: `ConvertToBinary ()`

- **OffsetValue**

  The **OffsetValue** function adds or subtracts *Value* from the value of the migrated object, and then writes the result back into the registry value on the destination computer. For example, if the migrated object is a DWORD with a value of `14`, and the *Value* is **"-2"**, the registry value will be `12` on the destination computer.

Syntax: `OffsetValue(Value)`

Expand table

Setting	Required?	Value
Value	Yes	The string representation of a numeric value. It can be positive or negative. For example, `OffsetValue(2)`.

- **SetValueByTable**

The **SetValueByTable** function matches the value from the source computer to the source table. If the value is there, the equivalent value in the destination table is applied. If the value isn't there, or if the destination table has no equivalent value, the *DefaultValueOnError* is applied.

Syntax: `SetValueByTable(SourceTable,DestinationTable,DefaultValueOnError)`

Expand table

Setting	Required?	Value
SourceTable	Yes	A list of values separated by commas that are possible for the source registry values.
DestinationTable	No	A list of translated values separated by commas.
DefaultValueOnError	No	The value that is applied to the destination computer if either 1. The value for the source computer doesn't match *SourceTable* 2. *DestinationTable* has no equivalent value. If **DefaultValueOnError** is **NULL**, the value isn't changed on the destination computer.

- **KeepExisting**

The **KeepExisting** function can be used when there are conflicts on the destination computer. This function keeps (not overwrites) the specified attributes for the object that is on the destination computer.

Syntax: `KeepExisting("OptionString","OptionString","OptionString",…)`

Expand table

Setting	Required?	Value
OptionString	Yes	*OptionString* can be **Security**, **TimeFields**, or **FileAttrib**:*Letter*. One of each type of *OptionStrings* can be specified. Don't specify multiple *OptionStrings* with the same value. If multiple *OptionStrings* with the same value are specified, the right-most option of that type is kept. For example, don't specify **("FileAttrib:H", "FileAttrib:R")** because only Read-only is evaluated. Instead specify **("FileAttrib:HR")** and both Hidden and Read-only attributes are kept on the destination computer. ○ **Security**: Keeps the destination object's security descriptor if it exists. ○ **TimeFields**: Keeps the destination object's time stamps. This parameter is for files only. ○ **FileAttrib:<Letter>**: Keeps the destination object's attribute value, either **ON** or **OFF**, for the specified set of file attributes. This parameter is for files only. The following are case-insensitive, but USMT will ignore any values that are invalid, repeated, or if there's a space after **FileAttrib:**. Any combination of the following attributes can be specified: ○ **A** = Archive ○ **C** = Compressed ○ **E** = Encrypted ○ **H** = Hidden ○ **I** = Not Content Indexed ○ **O** = Offline ○ **R** = Read-Only ○ **S** = System ○ **T** = Temporary

- **MergeMultiSzContent**

  The **MergeMultiSzContent** function merges the **MULTI-SZ** content of the registry values that are enumerated by the parent **<ObjectSet>** element with the content of the equivalent registry values that already exist on the destination computer. `Instruction` and `String` either remove or add content to the resulting **MULTI-SZ**. Duplicate elements are removed.

  Syntax: `MergeMultiSzContent (Instruction,String,Instruction,String,…)`

Expand table

Setting	Required?	Value
Instruction	Yes	Can be one of the following values:

Setting	Required?	Value
		◦ **Add**. Adds the corresponding String to the resulting MULTI-SZ if it isn't already there. ◦ **Remove**. Removes the corresponding String from the resulting MULTI-SZ.
String	Yes	The string to be added or removed.

- **MergeDelimitedContent**

  The **MergeDelimitedContent** function merges the content of the registry values that are enumerated by the parent **<ObjectSet>** element with the content of the equivalent registry values that already exist on the destination computer. The content is considered a list of elements separated by one of the characters in the Delimiters parameter. Duplicate elements are removed.

  Syntax: `MergeDelimitedContent(Delimiters,Instruction,String,…)`

Expand table

Setting	Required?	Value
Delimiters	Yes	A single character that is used to separate the content of the object that is being processed. The content is considered as a list of elements that is separated by the *Delimiters*. For example, `"."` separates the string based on a period.
Instruction	Yes	Can be one of the following values: ◦ **Add**: Adds *String* to the resulting MULTI-SZ if it isn't already there. ◦ **Remove**: Removes *String* from the resulting MULTI-SZ.
String	Yes	The string to be added or removed.

<description>

The **<description>** element defines a description for the component but doesn't affect the migration.

- **Number of occurrences:** zero or one
- **Parent elements:** <component>
- **Child elements:** none

Syntax:

```XML
<description>ComponentDescription</description>
```

Expand table

Setting	Required?	Value
ComponentDescription	Yes	The description of the component.

The following code sample shows how the <description> element defines the "My custom component" description:

```XML
<description>My custom component<description>
```

<destinationCleanup>

The **<destinationCleanup>** element deletes objects, such as files and registry keys, from the destination computer before applying the objects from the source computer. This element is evaluated only when the **LoadState** tool is run on the destination computer. That is, this element is ignored by the **ScanState** tool.

> **Important**
>
> Use this option with extreme caution because it will delete objects from the destination computer.

For each **<destinationCleanup>** element, there can be multiple **<objectSet>** elements. A common use for this element is if there's a missing registry key on the source computer but the component still needs to be migrated. In this case, all of the component's registry keys can be deleted before migrating the source registry keys. Deleting all of the component's registry keys ensures that if there's a missing key on the source computer, it will also be missing on the destination computer.

- **Number of occurrences:** Unlimited
- **Parent elements:** <rules>
- **Child elements:** <objectSet> (The destination computer deletes all child elements.)

Syntax:

```XML
<destinationCleanup filter=ScriptInvocation>
</destinationCleanup>
```

Expand table

Setting	Required?	Value
filter	Yes	A script followed by any number of string arguments that are separated by a comma and enclosed in parenthesis. For example, `MyScripts.AScript ("Arg1","Arg2")`. The script is called for each object that is enumerated by the object sets in the include rule. The filter script returns a Boolean value. If the return value is **TRUE**, the object is migrated. If it's **FALSE**, it isn't migrated.

For example:

```XML
<destinationCleanup>
   <objectSet>
      <pattern type="Registry">HKCU\Software\Lotus\123\99.0\DDE
Preferences\* [*]</pattern>
      <pattern type="Registry">HKCU\Software\Lotus\123\99.0\Find
Preferences\* [*]</pattern>
   </objectSet>
</destinationCleanup>
```

<detect>

Although the <**detect**> element is still supported, Microsoft recommends no longer using the <**detect**> element because it might be deprecated in future versions of USMT. If the <**detect**> element is deprecated, it would require a rewrite of any scripts that use the <**detect**> element. Instead, Microsoft recommends using the <detection> element. The <**detection**> element allows for more clearly formulated complex Boolean statements

The <**detect**> element can be used to determine if the component is present on a system. If all child <**detect**> elements within a <**detect**> element resolve to **TRUE**, then the <**detect**> element resolves to **TRUE**. If any child <**detect**> elements resolve to

FALSE, then their parent **<detect>** element resolves to **FALSE**. If there's no **<detect>** element section, then USMT assumes that the component is present.

For each **<detect>** element there can be multiple children **<condition>** or **<objectSet>** elements, which are logically joined by an **OR** operator. If at least one **<condition>** or **<objectSet>** element evaluates to **TRUE**, then the **<detect>** element evaluates to **TRUE**.

- **Number of occurrences:** unlimited
- **Parent elements:** <detects>, <namedElements>
- **Required child elements:** <condition>
- **Optional child elements:** <objectSet>

Syntax:

```XML
<detect name="ID" context="User|System|UserAndSystem">
</detect>
```

Expand table

Setting	Required?	Value
name	Yes, when **<detect>** is a child to **<namedElements>** No, when **<detect>** is a child to <detects>	When *ID* is specified, any child elements aren't processed. Instead, any other **<detect>** elements with the same name that are declared within the **<namedElements>** element are processed.

Setting	Required?	Value
context	No (default = UserAndSystem)	Defines the scope of this parameter, which is whether to process this component in the context of the specific user, across the entire operating system, or both. The largest possible scope is set by the component element. For example, if a **<component>** element has a context of **User**, and a **<rules>** element had a context of **UserAndSystem**, then the **<rules>** element would act as though it had a context of **User**. If the **<rules>** element had a context of **System**, it would act as though the **<rules>** element weren't there. • **User**: Evaluates the variables for each user. • **System**: Evaluates the variables only once for the system. • **UserAndSystem**: Evaluates the variables for the entire operating system and each user.

For examples, see the examples for <detection>.

<detects>

Although the **<detects>** element is still supported, Microsoft recommends no longer using the **<detects>** element because it might be deprecated in future versions of USMT. If the **<detects>** element is deprecated, it would require a rewrite of any scripts that use the **<detects>** element. Instead, Microsoft recommends using the **<detection>** element if the parent element is **<role>** or **<namedElements>**, or use the **<conditions>** element if the parent element is **<rules>**. The **<detection>** element allows for more clearly formulated complex Boolean statements and the **<conditions>** element allows for formulation of complex Boolean statements.

The **<detects>** element is a container for one or more **<detect>** elements. If all of the child **<detect>** elements within a **<detects>** element resolve to **TRUE**, then **<detects>** resolves to **TRUE**. If any of the child **<detect>** elements resolve to **FALSE**, then **<detects>** resolves to **FALSE**. To prevent the **<detects>** element to be written within a component, create the **<detects>** element under the **<namedElements>** element, and then refer to it. If there's no **<detects>** element section, then USMT assumes that the component is present. The results from each **<detects>** element are joined together by the **OR** operator to form the rule used to detect the parent element.

Syntax:

```
XML
```

```
<detects name="ID" context="User|System|UserAndSystem">
</detects>
```

- **Number of occurrences:** Unlimited.
- **Parent elements:** <role>, <rules>, <namedElements>
- **Required child elements:** <detect>

Expand table

Setting	Required?	Value
name	Yes, when <detects> is a child to **<namedElements>** No, when <detects> is a child to **<role>** or **<rules>**	When *ID* is specified, no child **<detect>** elements are processed. Instead, any other **<detects>** elements with the same name that are declared within the **<namedElements>** element are processed.
context	No (default = UserAndSystem)	Defines the scope of this parameter: whether to process this component in the context of the specific user, across the entire operating system, or both. The largest possible scope is set by the **<component element>**. For example, if a **<component>** element has a context of **User** and a **<rules>** element had a context of **UserAndSystem**, then the **<rules>** element would act as though it had a context of **User**. If the **<rules>** element had a context of **System**, it would act as though the **<rules>** element weren't there. • **User**: Evaluates the variables for each user. • **System**: Evaluates the variables only once for the system. • **UserAndSystem**: Evaluates the variables for the entire operating system and each user. The context parameter is ignored for **<detects>** elements that are inside **<rules>** elements.

The following example is from the `MigApp.xml` file.

XML

```
<detects>
   <detect>
```

```
<condition>MigXmlHelper.DoesFileVersionMatch("%Lotus123InstPath%\123w.exe","
ProductVersion","9.*")</condition>
   </detect>
   <detect>

<condition>MigXmlHelper.DoesFileVersionMatch("%SmartSuiteInstPath%\smartctr.
exe","ProductVersion","99.*")</condition>
   </detect>
</detects>
```

<detection>

The **<detection>** element is a container for one **<conditions>** element. The result of the child **<condition>** elements, located underneath the **<conditions>** element, determines the result of this element. For example, if all of the child **<conditions>** elements within the **<detection>** element resolve to **TRUE**, then the **<detection>** element resolves to **TRUE**. If any of the child **<conditions>** elements resolve to **FALSE**, then the **<detection>** element resolves to **FALSE**.

In addition, the results from each **<detection>** section within the **<role>** element are joined together by the **OR** operator to form the detection rule of the parent element. That is, if one of the **<detection>** sections resolve to **TRUE**, then the **<role>** element is processed. Otherwise, the **<role>** element isn't processed.

Use the **<detection>** element under the **<namedElements>** element to not write within a component. Then include a matching **<detection>** section under the **<role>** element to control whether the component is migrated. If there isn't a **<detection>** section for a component, then USMT assumes that the component is present.

- **Number of occurrences:** Unlimited.
- **Parent elements:** <role>, <namedElements>
- **Child elements:** <conditions>

Syntax:

XML

```
<detection name="ID" context="User|System|UserAndSystem">
</detection>
```

Expand table

Setting	Required?	Value
name	• Yes, when **<detection>** is declared under **<namedElements>** • Optional, when declared under **<role>**	If declared, the content of the **<detection>** element is ignored and the content of the **<detection>** element with the same name that is declared in the **<namedElements>** element is evaluated.
context	No, default = UserAndSystem	Defines the scope of this parameter: whether to process this component in the context of the specific user, across the entire operating system, or both. • **User**: Evaluates the component for each user. • **System**: Evaluates the component only once for the system. • **UserAndSystem**: Evaluates the component for the entire operating system and each user.

For example:

XML

```xml
<detection name="AdobePhotoshopCS">
   <conditions>

<condition>MigXmlHelper.DoesObjectExist("Registry","HKCU\Software\Adobe\Photoshop\8.0")</condition>

<condition>MigXmlHelper.DoesFileVersionMatch("%PhotoshopSuite8Path%\Photoshop.exe","FileVersion","8.*")</condition>
   </conditions>
</detection>
```

and

XML

```xml
<role role="Settings">
   <detection>
      <conditions>

<condition>MigXmlHelper.DoesFileVersionMatch("%QuickTime5Exe%","ProductVersion","QuickTime 5.*")</condition>

<condition>MigXmlHelper.DoesFileVersionMatch("%QuickTime5Exe%","ProductVersion","QuickTime 6.*")</condition>
      </conditions>
   </detection>
```

<displayName>

The **<displayName>** element is a required field within each **<component>** element.

- **Number of occurrences:** once for each component
- **Parent elements:** <component>
- **Child elements:** none

Syntax:

```XML
<displayName _locID="ID">ComponentName</displayName>
```

Expand table

Setting	Required?	Value
locID	No	This parameter is for internal USMT use. Don't use this parameter.
ComponentName	Yes	The name for the component.

For example:

```XML
<displayName>Command Prompt settings</displayName>
```

<environment>

The **<environment>** element is a container for **<variable>** elements in which variables can be defined for use in an **.xml** file. All environment variables defined this way are private. That is, they're available only for their child components and the component in which they were defined. For two example scenarios, see Examples.

- **Number of occurrences:** unlimited
- **Parent elements:** <role>, <component>, <namedElements>
- **Required child elements:** <variable>
- **Optional child elements:** <conditions>

Syntax:

```XML
<environment name="ID" context="User|System|UserAndSystem">
</environment>
```

Expand table

Setting	Required?	Value
name	Yes, when **<environment>** is a child of **<namedElements>** No, when **<environment>** is a child of **<role>** or **<component>**	When declared as a child of the **<role>** or **<component>** elements, if *ID* is declared, USMT ignores the content of the **<environment>** element and the content of the **<environment>** element with the same name declared in the **<namedElements>** element is processed.
context	No (default = UserAndSystem)	Defines the scope of this parameter: whether to process this component in the context of the specific user, across the entire operating system, or both. The largest possible scope is set by the **<component>** element. For example, if a **<component>** element has a context of **User** and a **<rules>** element had a context of **UserAndSystem**, then the **<rules>** element would act as though it had a context of **User**. If the **<rules>** element had a context of **System**, it would act as though **<rules>** weren't there. • **User**: Evaluates the variables for each user. • **System**: Evaluates the variables only once for the system. • **UserAndSystem**: Evaluates the variables for the entire operating system and each user.

Examples

Example scenario 1

In this scenario, generate the location of objects at run time depending on the configuration of the destination computer. For example, if an application writes data in the directory where the application is installed, and users can install the application anywhere on the computer. If the application writes a registry value

`hklm\software\companyname\install [path\]` and then updates this value with the location where the application is installed, then the only way to migrate the required data correctly is to define an environment variable. For example:

```XML
<environment>
   <variable name="INSTALLPATH">

<script>MigXmlHelper.GetStringContent("Registry","\software\companyname\inst
all [path]")</script>
   </variable>
</environment>
```

Then an include rule can be used as follows. Any of the <script> functions can be used to perform similar tasks.

```XML
<include>
   <objectSet>
      <pattern type="File">%INSTALLPATH%\ [*.xyz]</pattern>
   </objectSet>
</include>
```

Second, registry values can be filtered to contain the data that is needed. The following example extracts the first string (before the separator "`,`") in the value of the registry `Hklm\software\companyname\application\ [Path\]`.

```XML
<environment>
   <variable name="APPPATH">
        <objectSet>
           <content filter='MigXmlHelper.ExtractDirectory (",", "1")'>
             <objectSet>
                <pattern
type="Registry">Hklm\software\companyname\application\ [Path]</pattern>
              </objectSet>
            </content>
        </objectSet>
    </variable>
</environment>
```

Example scenario 2

In this scenario, five files named `File1.txt`, `File2.txt`, and so on, need to be migrated from `%SYSTEMDRIVE%\data\userdata\dir1\dir2\`. To migrate these files, the following **<include>** rule must be in an **.xml** file:

XML

```xml
<include>
   <objectSet>
      <pattern type="File">%SYSTEMDRIVE%\data\userdata\dir1\dir2 [File1.txt]
</pattern>
      <pattern type="File">%SYSTEMDRIVE%\data\userdata\dir1\dir2 [File2.txt]
</pattern>
      <pattern type="File">%SYSTEMDRIVE%\data\userdata\dir1\dir2 [File3.txt]
</pattern>
      <pattern type="File">%SYSTEMDRIVE%\data\userdata\dir1\dir2 [File4.txt]
</pattern>
      <pattern type="File">%SYSTEMDRIVE%\data\userdata\dir1\dir2 [File5.txt]
</pattern>
   </objectSet>
</include>
```

Instead of typing the path five times, create a variable for the location as follows:

XML

```xml
<environment>
   <variable name="DATAPATH">
      <text>%SYSTEMDRIVE%\data\userdata\dir1\dir2 </text>
      </variable>
</environment>
```

Then, specify the variable in an **<include>** rule as follows:

XML

```xml
<include>
   <objectSet>
      <pattern type="File">%DATAPATH% [File1.txt]</pattern>
      <pattern type="File">%DATAPATH% [File2.txt]</pattern>
      <pattern type="File">%DATAPATH% [File3.txt]</pattern>
      <pattern type="File">%DATAPATH% [File4.txt]</pattern>
      <pattern type="File">%DATAPATH% [File5.txt]</pattern>
   </objectSet>
</include>
```

<exclude>

The **<exclude>** element determines what objects aren't migrated, unless there's a more specific **<include>** element that migrates an object. If there's an **<include>** and **<exclude>** element for the same object, the object is included. For each **<exclude>** element, there can be multiple child **<objectSet>** elements.

- **Number of occurrences:** Unlimited
- **Parent elements:** <rules>
- **Child elements:** <objectSet>
- **Helper functions:** The following <exclude> filter functions can be used with this element: `CompareStringContent`, `IgnoreIrrelevantLinks`, `AnswerNo`, `NeverRestore`, and `SameRegContent`.

Syntax:

XML

```
<exclude filter="ScriptInvocation">
</exclude>
```

Expand table

Setting	Required?	Value
filter	No (default = No)	A script followed by any number of string arguments that are separated by a comma and enclosed in parenthesis. For example, `MyScripts.AScript("Arg1","Arg2")`. The script is called for each object that is enumerated by the object sets in the include rule. The filter script returns a Boolean value. If the return value is **TRUE**, the object is migrated. If it's **FALSE**, it isn't migrated.

For example, from the `MigUser.xml` file:

XML

```
<exclude>
   <objectSet>
      <pattern type="File">%CSIDL_MYMUSIC%\* [*]</pattern>
      <pattern type="File">%CSIDL_MYPICTURES%\* [*]</pattern>
      <pattern type="File">%CSIDL_MYVIDEO%\* [*]</pattern>
   </objectSet>
</exclude>
```

<excludeAttributes>

The **<excludeAttributes>** element can be used to determine which parameters associated with an object aren't migrated. If there are conflicts between the **<includeAttributes>** and **<excludeAttributes>** elements, the most specific pattern determines the patterns that aren't migrated. If an object doesn't have an **<includeAttributes>** or **<excludeAttributes>** element, then all of its parameters are migrated.

- **Number of occurrences:** Unlimited
- **Parent elements:** <rules>
- **Child elements:** <objectSet>

Syntax:

XML

```xml
<excludeAttributes attributes="Security|TimeFields|Security,TimeFields">
</excludeAttributes>
```

Expand table

Setting	Required?	Value
attributes	Yes	Specifies the attributes to be excluded. Either one of the following or both can be specified. If specifying both, they need to be separated by quotes. For example, `"Security","TimeFields"`: • Security can be one of Owner, Group, DACL, or SACL. • TimeFields can be one of CreationTime, LastAccessTime and LastWrittenTime

Example:

XML

```xml
<migration urlid="http://www.microsoft.com/migration/1.0/migxmlext/miguser">
<!-- This component migrates the files in the Video folder -->
   <component type="System" context="System">
      <displayName>System Data</displayName>
         <role role="Data">
            <rules>
<!-- Include all of the text files, which are immediately in the drive where
the operating system is installed -->
```

```
                <include>
                   <objectSet>
                      <pattern type="File">%SYSTEMDRIVE%\ [*.txt]</pattern>
                   </objectSet>
                </include>
<!-- Exclude the time stamps from the text file starting with the letter a -
->
                <excludeAttributes attributes="TimeFields">
                   <objectSet>
                      <pattern type="File">%SYSTEMDRIVE%\ [a*.txt]</pattern>
                   </objectSet>
                </excludeAttributes>
<!-- include the time stamps from the text file aa.txt -->
                <includeAttributes attributes="TimeFields">
                   <objectSet>
                      <pattern type="File">%SYSTEMDRIVE%\ [aa.txt]</pattern>
                   </objectSet>
                </includeAttributes>
<!-- Logoff the user after LoadState successfully completed. -->
                <externalProcess when="post-apply">
                   <commandLine>
                      logoff
                   </commandLine>
                </externalProcess>
         </rules>
   </role>
<!-- Migrate
   all doc files from the system
   all power point files
   all visio design files
   all my c++ program files -->
   <extensions>
      <extension>DOC</extension>
      <extension>PPT</extension>
      <extension>VXD</extension>
      <extension>PST</extension>
      <extension>CPP</extension>
   </extensions>
</component>
</migration>
```

<extensions>

The <extensions> element is a container for one or more <extension> elements.

- **Number of occurrences:** zero or one
- **Parent elements:** <component>
- **Required child elements:** <extension>

Syntax:

```XML
<extensions>
</extensions>
```

<extension>

The <extension> element can be used to specify documents of a specific extension.

- **Number of occurrences:** unlimited
- **Parent elements:** <extensions>
- **Child elements:** none

Syntax:

```XML
<extension>FilenameExtension</extension>
```

Expand table

Setting	Required?	Value
FilenameExtension	Yes	A file name extension.

For example, to migrate all *.doc files from the source computer, specifying the following code under the **<component>** element:

```XML
<extensions>
        <extension>doc</extension>
<extensions>
```

is the same as specifying the following code below the **<rules>** element:

```XML
<include>
        <objectSet>
                <script>MigXmlHelper.GenerateDrivePatterns ("* [*.doc]",
```

```
"Fixed")</script>
        </objectSet>
</include>
```

For another example of how to use the <extension> element, see the example for <excludeAttributes>.

<externalProcess>

The <externalProcess> element can be used to run a command line during the migration process. For example, a run a command might need to run after the **LoadState** process completes.

- **Number of occurrences:** Unlimited
- **Parent elements:** <rules>
- **Required child elements:** <commandLine>

Syntax:

XML

```
<externalProcess when="pre-scan|scan-success|post-scan|pre-apply|apply-success|post-apply">
</externalProcess>
```

Expand table

Setting	Required?	Value
when	Yes	Indicates when the command line should be run. This value can be one of the following values: • **pre-scan** before the scanning process begins. • **scan-success** after the scanning process finishes successfully. • **post-scan** after the scanning process finished, whether it was successful or not. • **pre-apply** before the apply process begins. • **apply-success** after the apply process finishes successfully. • **post-apply** after the apply process finished, whether it was successful or not.

For an example of how to use the <externalProcess> element, see the example for <excludeAttributes>.

<icon>

This element is an internal USMT element. Don't use this element.

<include>

The **<include>** element determines what to migrate, unless there's a more specific <exclude> rule. A script can be specified to be more specific to extend the definition of what want needs to be collected. For each **<include>** element, there can be multiple **<objectSet>** elements.

- **Number of occurrences:** Unlimited
- **Parent elements:** <rules>
- **Required child element:** <objectSet>
- **Helper functions:** The following <include> filter functions can be used with this element: `CompareStringContent`, `IgnoreIrrelevantLinks`, `AnswerNo`, and `NeverRestore`.

Syntax:

XML

```
<include filter="ScriptInvocation">
</include>
```

Expand table

Setting	Required?	Value
filter	No. If this parameter isn't specified, then all patterns that are inside the child **<objectSet>** element are processed.	A script followed by any number of string arguments that are separated by a comma and enclosed in parenthesis. For example, `MyScripts.AScript ("Arg1","Arg2")`. The script is called for each object that is enumerated by the object sets in the **<include>** rule. The filter script returns a Boolean value. If the return value is **TRUE**, the object is migrated. If it's **FALSE**, it isn't migrated.

The following example is from the `MigUser.xml` file:

XML

```
<component type="Documents" context="User">
   <displayName _locID="miguser.myvideo">My Video</displayName>
      <paths>
         <path type="File">%CSIDL_MYVIDEO%</path>
      </paths>
      <role role="Data">
         <detects>
            <detect>

<condition>MigXmlHelper.DoesObjectExist("File","%CSIDL_MYVIDEO%")
</condition>
            </detect>
         </detects>
         <rules>
               <include filter='MigXmlHelper.IgnoreIrrelevantLinks()'>
                  <objectSet>
                     <pattern type="File">%CSIDL_MYVIDEO%\* [*]</pattern>
                  </objectSet>
               </include>
               <merge script="MigXmlHelper.DestinationPriority()">
                  <objectSet>
                     <pattern type="File">%CSIDL_MYVIDEO% [desktop.ini]
</pattern>
                  </objectSet>
            </merge>
         </rules>
      </role>
    </component>
```

<include> and <exclude> filter functions

The following functions return a Boolean value. They can be used to migrate certain objects based on when certain conditions are met.

- **AnswerNo**

  This filter always returns **FALSE**.

  Syntax: `AnswerNo ()`

- **CompareStringContent**

  Syntax: `CompareStringContent("StringContent","CompareType")`

Expand table

Setting	Required?	Value
StringContent	Yes	The string to check against.
CompareType	Yes	A string. Use one of the following values: ○ **Equal** (case insensitive). The function returns **TRUE** if the string representation of the current object that is processed by the migration engine is identical to `StringContent`. ○ **NULL or any other value**. The function returns **TRUE** if the string representation of the current object that is processed by the migration engine doesn't match `StringContent`.

- **IgnoreIrrelevantLinks**

  This filter screens out the **.lnk** files that point to an object that isn't valid on the destination computer. The screening takes place on the destination computer, so all **.lnk** files are saved to the store during **ScanState**. Then they're screened out when the **LoadState** tool runs.

  Syntax: `IgnoreIrrelevantLinks ()`

  For example:

  XML

  ```xml
  <include filter='MigXmlHelper.IgnoreIrrelevantLinks()'>
       <objectSet>
            <pattern type="File">%CSIDL_COMMON_VIDEO%\* [*]</pattern>
       </objectSet>
  </include>
  ```

- **NeverRestore**

  This function can be used to collect the specified objects from the source computer but then not migrate the objects to the destination computer. When run with the **ScanState** tool, this function evaluates to **TRUE**. When run with the **LoadState** tool, this function evaluates to **FALSE**. This function might be used to check an object's value on the destination computer but there's no intention to migrate the object to the destination.

  Syntax: `NeverRestore()`

  In the following example, HKCU\Control Panel\International [Locale] is included in the store, but it isn't migrated to the destination computer:

  XML

```
<include filter="MigXmlHelper.NeverRestore()">
   <objectSet>
      <pattern type="Registry">HKCU\Control Panel\International
[Locale]</pattern>
   </objectSet>
</include>
```

<includeAttributes>

The **<includeAttributes>** element can be used to determine whether certain parameters associated with an object are migrated along with the object itself. If there are conflicts between the **<includeAttributes>** and **<excludeAttributes>** elements, the most specific pattern determines which parameters are migrated. If an object doesn't have an **<includeAttributes>** or **<excludeAttributes>** element, then all of its parameters are migrated.

- **Number of occurrences:** unlimited
- **Parent elements:** <rules>
- **Child elements:** <objectSet>

Syntax:

XML

```
<includeAttributes attributes="Security|TimeFields|Security,TimeFields">
</includeAttributes>
```

Expand table

Setting	Required?	Value

attributes	Yes	Specifies the attributes to be included with a migrated object. Either one of the following or both can be specified. If specifying both, they need to be separated by quotes. For example, `"Security","TimeFields"`:
		• Security can be one of the following values:
		  ○ **Owner**: The owner of the object (SID).
		  ○ **Group**: The primary group for the object (SID).
		  ○ **DACL** (discretionary access control list): An access control list that is controlled by the owner of an object and that specifies the access particular users or groups can have to the object.
		  ○ **SACL** (system access control list): An ACL that controls the generation of audit messages for attempts to access a

Setting	Required?	Value

		securable object. The ability to get or set an object's SACL is controlled by a privilege typically held only by system administrators.
		• TimeFields can be one of the following values:
		  ○ **CreationTime**: Specifies when the file or directory was created.
		  ○ **LastAccessTime**: Specifies when the file is last read from, written to, or for executable files, run.
		  ○ **LastWrittenTime**: Specifies when the file is last written to, truncated, or overwritten.

For an example of how to use the **<includeAttributes>** element, see the example for <excludeAttributes>.

<library>

This element is an internal USMT element. Don't use this element.

<location>

The **<location>** element defines the location of the **<object>** element.

- **Number of occurrences:** once for each **<object>**
- **Parent elements:** <object>
- **Child elements:** <script>

Syntax:

```XML
<location type="typeID">ObjectLocation</location>
```

Expand table

Setting	Required?	Value
type	Yes	*typeID* can be Registry or File.
ObjectLocation	Yes	The location of the object.

The following example is from the `MigApp.xml` file:

```XML
<addObjects>
   <object>
      <location
type="Registry">%HklmWowSoftware%\Microsoft\Office\16.0\Common\Migration\Off
ice [UpgradeVersion]</location>
      <attributes>DWORD</attributes>
      <bytes>0B000000</bytes>
   </object>
   <object>
      <location
type="Registry">%HklmWowSoftware%\Microsoft\Office\16.0\Common\Migration\Off
ice [Lang]</location>
      <attributes>DWORD</attributes>
      <bytes>00000000</bytes>
   </object>
</addObjects>
```

<locationModify>

The **<locationModify>** element can be used to change the location and name of an object before the object is migrated to the destination computer. The **<locationModify>** element is processed only when the **LoadState** tool is run on the destination computer. In other words, this element is ignored by the **ScanState** tool. The **<locationModify>** element creates the appropriate folder on the destination computer if it doesn't already exist.

Number of occurrences: Unlimited

- **Parent elements:** <rules>
- **Required child element:** <objectSet>
- **Helper functions:** The following <locationModify> functions can be used with this element: `ExactMove`, `RelativeMove`, and `Move`.

Syntax:

```XML
<locationModify script="ScriptInvocation">
</locationModify>
```

Expand table

Setting	Required?	Value
script	Yes	A script followed by any number of string arguments that are separated by a comma and enclosed in parenthesis. For example, `MyScripts.AScript("Arg1","Arg2")`. The script is called for each object that is enumerated by the object sets in the include rule. The filter script returns a Boolean value. If the return value is **TRUE**, the object is migrated. If it's **FALSE**, it isn't migrated.

The following example is from the `MigApp.xml` file:

XML

```xml
<locationModify
script="MigXmlHelper.RelativeMove('%CSIDL_APPDATA%\Microsoft\Office','%CSIDL
_APPDATA%')">
   <objectSet>
      <pattern type="File">%CSIDL_APPDATA%\Microsoft\Office\ [Access10.pip]
</pattern>
   </objectSet>
</locationModify>
```

<locationModify> functions

The following functions change the location of objects as they're migrated when using the **<locationModify>** element. These functions are called for every object that the parent **<objectSet>** element is enumerating. The **<locationModify>** element creates the appropriate folder on the destination computer if it doesn't already exist.

- **ExactMove**

  The ExactMove function moves all of the objects that are matched by the parent **<objectSet>** element into the given *ObjectEncodedLocation*. This function can be used to move a single file to a different location on the destination computer. If the destination location is a node, all of the matching source objects are written to the node without any subdirectories. If the destination location is a leaf, the migration engine migrates all of the matching source objects to the same location. If a collision occurs, the normal collision algorithms apply.

  Syntax: `ExactMove(ObjectEncodedLocation)`

Expand table

Setting	Required?	Value
ObjectEncodedLocation	Yes	The destination location for all of the source objects.

For example:

```XML
<locationModify script="MigXmlHelper.ExactMove('HKCU\Keyboard Layout\Toggle [HotKey]')">
   <objectSet>
        <pattern type="Registry">HKCU\Keyboard Layout\Toggle []
</pattern>
   </objectSet>
</locationModify>
```

- **Move**

  The Move function moves objects to a different location on the destination computer. In addition, this function creates subdirectories that were above the longest CSIDL in the source object name.

  Syntax: `Move(DestinationRoot)`

Expand table

Setting	Required?	Value
DestinationRoot	Yes	The location where the source objects are moved. If needed, this function creates any subdirectories that were above the longest CSIDL in the source object name.

- **RelativeMove**

  The RelativeMove function can be used to collect and move data. Environment variables can be used in source and destination roots, but they might be defined differently on the source and destination computers.

  Syntax: `RelativeMove(SourceRoot,DestinationRoot)`

Expand table

Setting	Required?	Value
SourceRoot	Yes	The location where the objects are moved from. Any source objects that are enumerated by the parent **<objectSet>**

Setting	Required?	Value
		element that aren't in this location aren't moved.
DestinationRoot	Yes	The location where the source objects are moved to on the destination computer. If needed, this function creates any subdirectories that were above *SourceRoot*.

For example:

XML

```
<include>
   <objectSet>
      <pattern type="File">%CSIDL_COMMON_FAVORITES%\* [*]</pattern>
   <objectSet>
</include>
<locationModify
script="MigXmlHelper.RelativeMove('%CSIDL_COMMON_FAVORITES%','%CSIDL_COMMON_
FAVORITES%')">
     <objectSet>
          <pattern type="File">%CSIDL_COMMON_FAVORITES%\* [*]</pattern>
     </objectSet>
</locationModify>
```

<_locDefinition>

This element is an internal USMT element. Don't use this element.

<manufacturer>

The **<manufacturer>** element defines the manufacturer for the component, but doesn't affect the migration.

- **Number of occurrences:** zero or one
- **Parent elements:** <component>
- **Child elements:** none

Syntax:

XML

```
<manufacturer>Name</manufacturer>
```

Expand table

Setting	Required?	Value
Name	Yes	The name of the manufacturer for the component.

<merge>

The <**merge**> element determines what happens when a collision occurs. A collision is when an object that is migrated is already present on the destination computer. If this element isn't specified, the default behavior for the registry is for the source object to overwrite the destination object. The default behavior for files is for the source file to be renamed to `OriginalFileName(1).OriginalExtension`. This element specifies only what should be done when a collision occurs. It doesn't include objects. Therefore, for the objects to migrate, <**include**> rules must be specified along with the <**merge**> element. When an object is processed and a collision is detected, USMT selects the most specific merge rule. It then applies the rule to resolve the conflict. For example, if a <**merge**> rule `C:\* [*]` is set to <**sourcePriority**> and a <**merge**> rule `C:\subfolder\* [*]` is set to <**destinationPriority**>, then USMT would use the <**destinationPriority**> rule because it's the more specific.

For an example of this element, see Conflicts and precedence.

- **Number of occurrences:** Unlimited
- **Parent elements:** <rules>
- **Required child element:** <objectSet>
- **Helper functions:** The following <merge> functions can be used with this element: `SourcePriority`, `DestinationPriority`, `FindFilePlaceByPattern`, `LeafPattern`, `NewestVersion`, `HigherValue()`, and `LowerValue()`.

Syntax:

XML

```
<merge script="ScriptInvocation">
</merge>
```

Expand table

Setting	Required?	Value
script	Yes	A script followed by any number of string arguments that are separated by a comma and enclosed in parenthesis. For example, `MyScripts.AScript("Arg1","Arg2")`. The script is called for each object that is enumerated by the object sets in the **<include>** rule. The filter script returns a Boolean value. If the return value is **TRUE**, the object is migrated. If it's **FALSE**, it isn't migrated.

The following example is from the `MigUser.xml` file:

XML

```xml
<rules>
   <include filter='MigXmlHelper.IgnoreIrrelevantLinks()'>
      <objectSet>
         <pattern type="File">%CSIDL_MYVIDEO%\* [*]</pattern>
      </objectSet>
   </include>
   <merge script="MigXmlHelper.DestinationPriority()">
      <objectSet>
         <pattern type="File">%CSIDL_MYVIDEO% [desktop.ini]</pattern>
      </objectSet>
   </merge>
</rules>
```

<merge> functions

These functions control how collisions are resolved.

- **DestinationPriority**

  Specifies to keep the object that is on the destination computer and not migrate the object from the source computer.

  For example:

  XML

  ```xml
  <merge script="MigXmlHelper.DestinationPriority()">
       <objectSet>
            <pattern
  type="Registry">HKCU\Software\Microsoft\Office\16.0\PhotoDraw\
  [MyPictures]</pattern>
            <pattern
  type="Registry">HKCU\Software\Microsoft\Office\16.0\PhotoDraw\Settings\
  [PicturesPath]</pattern>
            <pattern
  ```

```
type="Registry">HKCU\Software\Microsoft\Office\16.0\PhotoDraw\Settings\
[AdditionalPlugInPath]</pattern>
     </objectSet>
</merge>
```

- **FindFilePlaceByPattern**

  The FindFilePlaceByPattern function saves files with an incrementing counter when a collision occurs. It's a string that contains one of each construct: **<F>**, **<E>**, **<N>** in any order.

  Syntax: `FindFilePlaceByPattern(FilePattern)`

Expand table

Setting	Required?	Value
FilePattern	Yes	◦ **<F>** is replaced by the original file name. ◦ **<N>** is replaced by an incrementing counter until there's no collision with the objects on the destination computer. ◦ **<E>** is replaced by the original file name extension. For example, `<F> (<N>).<E>` changes the source file `MyDocument.doc` into `MyDocument (1).doc` on the destination computer.

- **NewestVersion**

  The NewestVersion function resolves conflicts on the destination computer based on the version of the file.

  Syntax: `NewestVersion(VersionTag)`

Expand table

Setting	Required?	Value
VersionTag	Yes	The version field that is checked. This field can be `FileVersion` or `ProductVersion`. The file with the highest *VersionTag* version determines which conflicts are resolved based on the file's version. For example, if `Myfile.txt` contains FileVersion 1 and the same file on the destination computer contains FileVersion 2, the file on destination remains.

- **HigherValue()**

This function can be used for merging registry values. The registry values are evaluated as numeric values, and the one with the higher value determines which registry values are merged.

- **LowerValue()**

  This function can be used for merging registry values. The registry values are evaluated as numeric values and the one with the lower value determines which registry values are merged.

- **SourcePriority**

  Specifies to migrate the object from the source computer, and to delete the object that is on the destination computer.

  For example:

```xml
<merge script="MigXmlHelper.SourcePriority()">
 <objectSet>
   <pattern
type="Registry">%HklmWowSoftware%\Microsoft\Office\14.0\Common\Migratio
n\Publisher [UpgradeVersion]</pattern>
   <pattern
type="Registry">%HklmWowSoftware%\Microsoft\Office\15.0\Common\Migratio
n\Publisher [UpgradeVersion]</pattern>
   <pattern
type="Registry">%HklmWowSoftware%\Microsoft\Office\16.0\Common\Migratio
n\Publisher [UpgradeVersion]</pattern>
 </objectSet>
</merge>
```

<migration>

The **<migration>** element is the single root element of a migration **.xml** file and is required. Each **.xml** file must have a unique migration urlid. The urlid of each file that is specified on the command line must be unique. The urlids must be unique because USMT uses the urlid to define the components within the file.

- **Number of occurrences:** one
- **Parent elements:** none
- **Required child elements:** <component>
- **Optional child elements:** <library>, <namedElements>

Syntax:

```XML
<migration urlid="*UrlID/*Name">
</migration>
```

Expand table

Setting	Required?	Value
urlid	Yes	*UrlID* is a string identifier that uniquely identifies this **.xml** file. This parameter must be a no-colon-name as defined by the XML Namespaces specification. Each migration **.xml** file must have a unique urlid. If two migration **.xml** files have the same urlid, the second **.xml** file that is specified on the command line isn't processed. For more information about XML Namespaces, see Use XML Namespaces.
Name	No	Although not required, it's good practice to use the name of the **.xml** file.

The following example is from the `MigApp.xml` file:

```XML
<migration urlid="http://www.microsoft.com/migration/1.0/migxmlext/migapp">
</migration>
```

MigXMLHelper.FileProperties

This filter helper function can be used to filter the migration of files based on file size and date attributes.

Expand table

Helper Function	MigXMLHelper.FileProperties (property, operator, valueToCompare)
Property	filesize, dateCreated, dateModified, dateAccessed
Operator	range, neq, lte, lt, eq, gte, gt
valueToCompare	The value that is being compared. For example: Date: "2023/05/15-2020/05/17", "2023/05/15" Size: A numeral with B, KB, MB, or GB at the end. "5GB", "1KB-1MB"

XML

```
<component context="System"  type="Application">
<displayName>File_size</displayName>
<role role="Data">

   <rules>
        <include
filter='MigXmlHelper.FileProperties("dateAccessed","range","2023/05/15-
2020/05/17")'>
         <objectSet>
         <pattern type="File">%SYSTEMDRIVE%\DOCS\* [*]</pattern>
         </objectSet>
      </include>
   </rules>
</role>
</component>
```

<namedElements>

The **<namedElements>** element can be used to define named elements. These elements can be used in any component throughout the **.xml** file. For an example of how to use this element, see the `MigApp.xml` file.

Syntax:

XML

```
<namedElements>
</namedElements>
```

- **Number of occurrences:** Unlimited
- **Parent elements:** <migration>
- **Child elements:** <environment>, <rules>, <conditions>, <detection>, <detects>, <detect>

For an example of this element, see the `MigApp.xml` file.

<object>

The **<object>** element represents a file or registry key.

- **Number of occurrences:** Unlimited
- **Parent elements:** <addObjects>

- **Required child elements:** <location>, <attributes>
- **Optional child elements:** <bytes>

Syntax:

```XML
<object>
</object>
```

The following example is from the `MigApp.xml` file:

```XML
<addObjects>
   <object>
      <location
type="Registry">%HklmWowSoftware%\Microsoft\Office\16.0\Common\Migration\Off
ice [UpgradeVersion]</location>
      <attributes>DWORD</attributes>
      <bytes>0B000000</bytes>
   </object>
   <object>
      <location
type="Registry">%HklmWowSoftware%\Microsoft\Office\16.0\Common\Migration\Off
ice [Lang]</location>
      <attributes>DWORD</attributes>
      <bytes>00000000</bytes>
      </object>
</addObjects>
```

<objectSet>

The <**objectSet**> element contains a list of object patterns; for example, file paths, registry locations, and so on. Any child <**conditions**> elements are evaluated first. If all child <**conditions**> elements return **FALSE**, the <**objectSet**> element evaluates to an empty set. For each parent element, there can be only multiple <**objectSet**> elements.

- **Number of occurrences:** Unlimited
- **Parent elements:** <variable>, <content>, <include>, <exclude>, <merge>, <contentModify>, <locationModify>, <destinationCleanup>, <includeAttributes>, <excludeAttributes>, <unconditionalExclude>, <detect>
- **Required child elements:** either <script> or <pattern>

- **Optional child elements:** <content>, <conditions>, <condition>

Syntax:

```XML
<objectSet>
</objectSet>
```

The following example is from the `MigUser.xml` file:

```XML
<component type="Documents" context="User">
   <displayName _locID="miguser.mymusic">My Music</displayName>
      <paths>
         <path type="File">%CSIDL_MYMUSIC%</path>
      </paths>
   <role role="Data">
      <detects>
      <detect>
         <condition>MigXmlHelper.DoesObjectExist("File","%CSIDL_MYMUSIC%")
</condition>
      </detect>
   </detects>
   <rules>
      <include filter='MigXmlHelper.IgnoreIrrelevantLinks()'>
         <objectSet>
            <pattern type="File">%CSIDL_MYMUSIC%\* [*]</pattern>
         </objectSet>
      </include>
      <merge script="MigXmlHelper.DestinationPriority()">
         <objectSet>
            <pattern type="File">%CSIDL_MYMUSIC%\ [desktop.ini]</pattern>
         </objectSet>
      </merge>
   </rules>
   </role>
</component>
```

<path>

This element is an internal USMT element. Don't use this element.

<paths>

This element is an internal USMT element. Don't use this element.

<pattern>

This element can be used to specify multiple objects. Multiple **<pattern>** elements can be used for each **<objectSet>** element and they're combined. If specifying files, Microsoft recommends using `GenerateDrivePatterns` with **<script>** instead. `GenerateDrivePatterns` is basically the same as a **<pattern>** rule, without the drive letter specification. For example, the following two lines of code are similar:

XML

```
<pattern type="File">C:\Folder\* [Sample.doc]</pattern>
<script>MigXmlHelper.GenerateDrivePatterns("\Folder\* [Sample.doc]","Fixed"
</script>
```

- **Number of occurrences:** Unlimited
- **Parent elements:** <objectSet>
- **Child elements:** none but *Path* [*object*] must be valid.

Syntax:

XML

```
<pattern type="typeID">Path [object]</pattern>
```

Expand table

Setting	Required?	Value
type	Yes	*typeID* can be Registry, File, or Ini. If *typeId* is Ini, then a space between *Path* and *object* isn't allowed. For example, the following format is correct when type="Ini": **<pattern type="Ini">%WinAmp5InstPath%\Winamp.ini|WinAmp[keeponscreen] </pattern>**
Path [*object*]	Yes	A valid registry or file path pattern, followed by at least one space, followed by brackets [] that contain the object to be migrated. • *Path* can contain the asterisk (`*`) wildcard character or can be an Recognized environment variables. The question mark can't be used as a wildcard character. `HKCU` and `HKLM` can be used to refer to `HKEY_CURRENT_USER` and `HKEY_LOCAL_MACHINE` respectively. • *Object* can contain the asterisk (`*`) wildcard character. However, the question mark can't be used as a wildcard character. For example: `C:\Folder\ [*]` enumerates all files in `C:\Folder` but no subfolders

Setting	Required?	Value
		of `C:\Folder`. **`C:\Folder* [*]`** enumerates all files and subfolders of `C:\Folder`. **`C:\Folder\ [*.mp3]`** enumerates all `.mp3` files in `C:\Folder`. **`C:\Folder\ [Sample.doc]`** enumerates only the `Sample.doc` file located in C:\Folder. **Note** If migrating a file that has a square bracket character ([or]) in the file name, a carrot (^) character must be inserted directly before the bracket for it to be valid. For example, if there's a file named "file].txt", `<pattern type="File">c:\documents\mydocs [file^].txt] </pattern>` must be specified instead of `<pattern type="File">c:\documents\mydocs [file].txt] </pattern>`.

For example:

- To migrate a single registry key:

```XML
<pattern
type="Registry">HKLM\Software\Microsoft\Windows\CurrentVersion\Internet
Settings\Cache [Persistent]</pattern>
```

- To migrate the `C:\EngineeringDrafts` folder and any subfolders from the C: drive:

```XML
<pattern type="File">C:\EngineeringDrafts\* [*]</pattern>
```

- To migrate only the `C:\EngineeringDrafts` folder, excluding any subfolders, from the C: drive:

  Reroute files and settings

- To migrate the `Sample.doc` file from `C:\EngineeringDrafts`:

```XML
```

```
<pattern type="File"> C:\EngineeringDrafts\ [Sample.doc]</pattern>
```

- To migrate the `Sample.doc` file from where ever it exists on the C: drive use pattern in the following way. If multiple files exist with the same name on the C: drive, then all of these files are migrated.

  XML

  ```
  <pattern type="File"> C:\* [Sample.doc] </pattern>
  ```

- For more examples of how to use this element, see Exclude files and settings, Reroute files and settings, Include files and settings, and Custom XML examples.

<processing>

This element can be used to run a script during a specific point within the migration process. Return values aren't expected from the scripts that are specified. If there are return values, they're ignored.

- **Number of occurrences:** unlimited
- **Parent elements:** <rules>
- **Required child element:** <script>

Syntax:

XML

```
<processing when="pre-scan|scan-success|post-scan|pre-apply|apply-
success|post-apply">
</processing>
```

Expand table

Setting	Required?	Value
when	Yes	Indicates when the script should be run. This value can be one of the following values: • **pre-scan** means before the scanning process begins. • **scan-success** means after the scanning process finishes successfully. • **post-scan** means after the scanning process finished, whether it was successful or not. • **pre-apply** means before the apply process begins.

Setting	Required?	Value
		• **apply-success** means after the apply process finishes successfully. • **post-apply** means after the apply process finished, whether it was successful or not.

<plugin>

This element is an internal USMT element. Don't use this element.

<role>

The **<role>** element is required in a custom **.xml** file. When the **<role>** element is specified, a concrete component can be created. The component is defined by the parameters specified at the **<component>** level, and with the role that is specified here.

- **Number of occurrences:** Each **<component>** can have one, two or three child **<role>** elements.
- **Parent elements:** <component>, <role>
- **Required child elements:** <rules>
- **Optional child elements:** <environment>, <detection>, <component>, <role>, <detects>, <plugin>

Syntax:

XML

```xml
<role role="Container|Binaries|Settings|Data">
</role>
```

Expand table

Setting	Required?	Value
role	Yes	Defines the role for the component. Role can be one of: • **Container** • **Binaries** • **Settings** • **Data** One of the following items can be specified:

Setting	Required?	Value

		1. Up to three **<role>** elements within a **<component>** - one "Binaries" role element, one "Settings" role element and one "Data" role element. These parameters don't change the migration behavior - their only purpose is to help categorize the settings that are migrating. These **<role>** elements can be nested, but each nested element must be of the same role parameter.
		2. One "Container" **<role>** element within a **<component>** element. In this case, any child **<rules>** elements can't be specified, only other **<component>** elements. And each child **<component>** element must have the same type as that of parent **<component>** element. For example:
		
		`<component context="UserAndSystem" type="Application">`
		`  <displayName _locID="migapp.msoffice2016">Microsoft Office 2016</displayName>`
		`  <environment name="GlobalEnv" />`
		`  <role role="Container">`
		`  <detection name="AnyOffice2016Version" />`
		`  <detection name="Word2016" />`
		`  <!--`
		` Office 2016 Common Settings`
		`  -->`
		`  <component context="UserAndSystem" type="Application">`

The following example is from the `MigUser.xml` file. For more examples, see the `MigApp.xml` file:

XML

```xml
<component type="System" context="User">
   <displayName _locID="miguser.startmenu">Start Menu</displayName>
   <paths>
      <path type="File">%CSIDL_STARTMENU%</path>
   </paths>
   <role role="Settings">
      <detects>
         <detect>

<condition>MigXmlHelper.DoesObjectExist("File","%CSIDL_STARTMENU%")
</condition>
         </detect>
      </detects>
   <rules>
      <include filter='MigXmlHelper.IgnoreIrrelevantLinks()'>
         <objectSet>
            <pattern type="File">%CSIDL_STARTMENU%\* [*]</pattern>
```

```
        </objectSet>
      </include>
      <merge script="MigXmlHelper.DestinationPriority()">
        <objectSet>
          <pattern type="File">%CSIDL_STARTMENU% [desktop.ini]</pattern>
          <pattern type="File">%CSIDL_STARTMENU%\* [*]</pattern>
        </objectSet>
      </merge>
    </rules>
    </role>
  </component>
```

<rules>

The <**rules**> element is required in a custom **.xml** file. This element contains rules that run during the migration if the parent <**component**> element is selected, unless the child <**conditions**> element, if present, evaluates to **FALSE**. For each <**rules**> element, there can be multiple child <**rules**> elements.

- **Number of occurrences:** unlimited
- **Parent elements:** <role>, <rules>, <namedElements>
- **Required child elements:** <include>
- **Optional child elements:** <rules>, <exclude>, <unconditionalExclude>,<merge>, <contentModify>, <locationModify>, <destinationCleanup>, <addObjects>, <externalProcess>, <processing>, <includeAttributes>, <excludeAttributes>, conditions, <detects>

Syntax:

XML

```
<rules name="ID" context="User|System|UserAndSystem">
</rules>
```

Expand table

Setting	Required?	Value
name	Yes, when <**rules**> is a child to <**namedElements**> No, when <**rules**> is a	When *ID* is specified, any child elements aren't processed. Instead, any other <**rules**> elements with the same name that are declared within <**namedElements**> are processed.

Setting	Required?	Value
	child to any other element	
context	No (default = UserAndSystem)	Defines the scope of this parameter - whether to process this component in the context of the specific user, across the entire operating system, or both. The largest possible scope is set by the component element. For example, if a **<component>** element has a context of **User** and a **<rules>** element had a context of **UserAndSystem**, then the **<rules>** element would act as though it has a context of **User**. If **<rules>** had a context of **System**, it would act as though **<rules>** wasn't there. • **User**: Evaluates the variables for each user. • **System**: Evaluates the variables only once for the system. • **UserAndSystem**: Evaluates the variables for the entire operating system and each user.

The following example is from the `MigUser.xml` file:

XML

```xml
<component type="Documents" context="User">
   <displayName _locID="miguser.mymusic">My Music</displayName>
      <paths>
         <path type="File">%CSIDL_MYMUSIC%</path>
      </paths>
   <role role="Data">
      <detects>
      <detect>
         <condition>MigXmlHelper.DoesObjectExist("File","%CSIDL_MYMUSIC%")
</condition>
      </detect>
   </detects>
   <rules>
      <include filter='MigXmlHelper.IgnoreIrrelevantLinks()'>
         <objectSet>
            <pattern type="File">%CSIDL_MYMUSIC%\* [*]</pattern>
         </objectSet>
      </include>
      <merge script="MigXmlHelper.DestinationPriority()">
         <objectSet>
            <pattern type="File">%CSIDL_MYMUSIC%\ [desktop.ini]</pattern>
         </objectSet>
      </merge>
   </rules>
```

```
    </role>
  </component>
```

<script>

The return value that is required by **<script>** depends on the parent element.

Number of occurrences: Once for <variable>, unlimited for <objectSet> and <processing>

Parent elements: <objectSet>, <variable>, <processing>

Child elements: none

Syntax and helper functions:

- General Syntax: `<script>ScriptWithArguments</script>`
- GetStringContent can be used when **<script>** is within **<variable>**.

  Syntax: `<script>MigXmlHelper.GetStringContent("ObjectType","EncodedLocationPattern", "ExpandContent")</script>`

  Example: `<script>MigXMLHelper.GetStringContent("Registry","HKLM\Software\MyApp\Installer [EXEPATH]")</script>`
- GenerateUserPatterns can be used when **<script>** is within **<objectSet>**.

  Syntax: `<script>MigXmlHelper.GenerateUserPatterns("ObjectType","EncodedLocationPattern","ProcessCurrentUser")</script>`

  Example: `<script>MigXmlHelper.GenerateUserPatterns ("File","%USERPROFILE%\*[*.doc]", "FALSE")</script>`
- GenerateDrivePatterns can be used when **<script>** is within **<objectSet>**.

  Syntax: `<script>MigXmlHelper.GenerateDrivePatterns("PatternSegment","DriveType")</script>`

Example: `<script>MigXmlHelper.GenerateDrivePatterns("* [sample.doc]", "Fixed")</script>`

- The Simple executing scripts can be used with **<script>** elements that are within **<processing>** elements: AskForLogoff, ConvertToShortFileName, KillExplorer, RemoveEmptyDirectories, RestartExplorer, RegisterFonts, StartService, StopService, SyncSCM.

  Syntax: `<script>MigXmlHelper.ExecutingScript</script>`

  Example: `<script>MigXmlHelper.KillExplorer()</script>`

Expand table

Setting	Required?	Value
ScriptWithArguments	Yes	A script followed by any number of string arguments that are separated by a comma and enclosed in parenthesis. For example, `MyScripts.AScript ("Arg1","Arg2")`. The script is called for each object that is enumerated by the object sets in the **<include>** rule. The filter script returns a Boolean value. If the return value is **TRUE**, the object is migrated. If it's **FALSE**, it isn't migrated. The return value that is required by **<script>** depends on the parent element. • When used within **<variable>**, the return value must be a string. • When used within **<objectSet>**, the return value must be a two-dimensional array of strings. • When used within **<location>**, the return value must be a valid location that aligns with the type attribute of **<location>**. For example, if <location type="File">, the child script element, if specified, must be a valid file location. **Note** If migrating a file that has a bracket character ([or]) in the file name, insert the carrot (^) character directly before the bracket for it to be valid. For example, if there's a file named "file].txt", specify `<pattern type="File">c:\documents\mydocs[file^].txt]</pattern>` instead of

Setting	Required?	Value
		`<pattern type="File">c:\documents\mydocs [file].txt]</pattern>`.

Examples:

To migrate the Sample.doc file from any drive on the source computer, use **<script>** as follows. If multiple files exist with the same name, all such files get migrated.

```XML
<script>MigXmlHelper.GenerateDrivePatterns("* [sample.doc]", "Fixed")
</script>
```

For more examples of how to use this element, see Exclude Files and Settings, Reroute Files and Settings, and Custom XML Examples.

<script> functions

The following functions can be used with the **<script>** element

- String and pattern generating functions
- Simple executing scripts

String and pattern generating functions

These functions return either a string or a pattern.

- **GetStringContent**

  GetStringContent can be used with **<script>** elements that are within **<variable>** elements. If possible, this function returns the string representation of the given object. Otherwise, it returns **NULL**. For file objects this function, always returns **NULL**.

  Syntax: `GetStringContent("ObjectType","EncodedLocationPattern", "ExpandContent")`

Expand table

Setting	Required?	Value
ObjectType	Yes	The type of object. Can be Registry or Ini (for an **.ini** file).
EncodedLocationPattern	Yes	◦ If type of object is Registry, EncodedLocationPattern must be a valid registry path. For example, `HKLM\SOFTWARE\MyKey[]`. ◦ If the type of object is Ini, then EncodedLocationPattern must be in the following format: **IniFilePath|SectionName[SettingName]**
ExpandContent	No (default=TRUE)	Can be **TRUE** or **FALSE**. If **FALSE**, then the given location isn't expanded before returned.

For example:

```XML
<variable name="MSNMessengerInstPath">
<script>MigXmlHelper.GetStringContent("Registry","%HklmWowSoftware%\Microsoft\MSNMessenger [InstallationDirectory]")</script>
</variable>
```

- **GenerateDrivePatterns**

The `GenerateDrivePatterns` function iterates all of the available drives and selects the ones that match the requested drive type. It then concatenates the selected drives with the end part of *PatternSegment* to form a full encoded file pattern. For example, if *PatternSegment* is `Path [file.txt]` and *DriveType* is `Fixed`, then the function generates `C:\Path [file.txt]`, and other patterns if there are fixed drives other than C:. Environment variables can't be specified with this function. `GenerateDrivePatterns` can be used with **<script>** elements that are within <objectSet> that are within **<include>**/**<exclude>**.

Syntax: `GenerateDrivePatterns("PatternSegment","DriveType")`

Expand table

Setting	Required?	Value
PatternSegment	Yes	The suffix of an encoded pattern. The value is concatenated with a drive specification, such as "c:", to form a complete

Setting	Required?	Value
		encoded file pattern. For example, "* [*.doc]". *PatternSegment* can't be an environment variable.
DriveType	Yes	The drive type for which the patterns are to be generated. One of the following items can be specified: ○ Fixed ○ CDROM ○ Removable ○ Remote

See the last component in the `MigUser.xml` file for an example of this element.

- **GenerateUserPatterns**

  The `GenerateUserPatterns` function iterates through all users that are being migrated, excluding the currently processed user if <**ProcessCurrentUser**> is **FALSE**, and expands the specified pattern in the context of each user. For example, if users A, B, and C have profiles in `C:\Users`, by calling `GenerateUserPattens('File','%userprofile% [*.doc]','TRUE')`, the helper function generates the following three patterns:

  - "C:\Users\A* [*.doc]"
  - "C:\Users\B* [*.doc]"
  - "C:\Users\C* [*.doc]"

  Syntax:
  `GenerateUserPatterns("ObjectType","EncodedLocationPattern","ProcessCurrentUser")`

Expand table

Setting	Required?	Value
ObjectType	Yes	Defines the object type. Can be File or Registry.
EncodedLocationPattern	Yes	The location pattern. Environment variables are allowed.
ProcessCurrentUser	Yes	Can be **TRUE** or **FALSE**. Indicates if the patterns should be generated for the current user.

Example:

If `GenerateUserPattens('File','%userprofile% [*.doc]','FALSE')` is called while USMT is processing user A, then this function only generates patterns for users B and C. This helper function can be used to build complex rules. For example, to migrate all `.doc` files from the source computer - but if user X isn't migrated, then don't migrate any of the `.doc` files from user X's profile.

The following example is example code for this scenario. The first **<rules>** element migrates all `.doc` files on the source computer except for those inside `C:\Users`. The second **<rules>** elements migrate all `.doc` files from `C:\Users` except for the `.doc` files in the profiles of the other users. Because the second **<rules>** element is processed in each migrated user context, the end result is the desired behavior. The end result is the one we expected.

XML

```xml
<rules context="System">
  <include>
    <objectSet>
      <script>MigXmlHelper.GenerateDrivePatterns ("* [*.doc]", "Fixed")
</script>
    </objectSet>
  </include>
  <exclude>
    <objectSet>
      <pattern type="File">%ProfilesFolder%\* [*.doc]</pattern>
    </objectSet>
  </exclude>
</rules>
<rules context="User">
  <include>
    <objectSet>
      <pattern type="File">%ProfilesFolder%\* [*.doc]</pattern>
    </objectSet>
  </include>
  <exclude>
    <objectSet>
      <script>MigXmlHelper.GenerateUserPatterns ("File","%userprofile%\*
[*.doc]", "FALSE")</script>
    </objectSet>
  </exclude>
</rules>
```

MigXmlHelper.GenerateDocPatterns

The `MigXmlHelper.GenerateDocPatterns` helper function invokes the document finder to scan the system for all files that can be migrated. It can be invoked in either **System** or **User** context to focus the scan.

Expand table

Setting	Required?	Value
ScanProgramFiles	No (default = FALSE)	Can be **TRUE** or **FALSE**. The *ScanProgramFiles* parameter determines whether or not the document finder scans the **Program Files** directory to gather registered file extensions for known applications. For example, when set to **TRUE** it discovers and migrates **.jpg** files under the Photoshop directory, if `.jpg` is a file extension registered to Photoshop.
IncludePatterns	No (default = TRUE)	Can be **TRUE** or **FALSE**. **TRUE** generates include patterns and can be added under the **<include>** element. **FALSE** generates exclude patterns and can be added under the **<exclude>** element.
SystemDrive	No (default = FALSE)	Can be **TRUE** or **FALSE**. If **TRUE**, restricts all patterns to the system drive.

XML

```
<!-- This component migrates data in user context -->
 <component type="Documents" context="User">
   <displayName>MigDocUser</displayName>
   <role role="Data">
     <rules>
       <include filter='MigXmlHelper.IgnoreIrrelevantLinks()'>
         <objectSet>
           <script>MigXmlHelper.GenerateDocPatterns ("false")</script>
         </objectSet>
       </include>
       <exclude>
         <objectSet>
          <script>MigXmlHelper.GenerateDocPatterns ("false", "false",
"false")</script>
         </objectSet>
       </exclude>
     </rules>
   </role>
 </component>
```

Simple executing scripts

The following scripts have no return value. The following errors can be used with **<script>** elements that are within **<processing>** elements

- **AskForLogoff()**. Prompts the user to sign out at the end of the migration. For example:

```XML
<processing when="apply-success">
  <script>MigXmlHelper.AskForLogoff()</script>
</processing>
```

- **ConvertToShortFileName(RegistryEncodedLocation)**. If *RegistryEncodedLocation* is the full path of an existing file, this function converts the file to its short file name and then it updates the registry value.

- **KillExplorer()**. Stops Explorer.exe for the current user context. Stopping Explorer.exe allows access to certain keys and files that are kept open when Explorer.exe is running. For example:

```XML
<processing when="pre-apply">
  <script>MigXmlHelper.KillExplorer()</script>
</processing>
```

- **RegisterFonts(FileEncodedLocation)**. Registers the given font or all of the fonts in the given directory. For example:

```XML
<processing when="apply-success">
  <script>MigXmlHelper.RegisterFonts("%CSIDL_COMMON_FONTS%")</script>
</processing>
```

- **RemoveEmptyDirectories (DirectoryEncodedPattern).** Deletes any empty directories that match *DirectoryEncodedPattern* on the destination computer.

- **RestartExplorer().** Restarts Explorer.exe at the end of the migration. For example:

```XML
<processing when="post-apply">
  <script>MigXmlHelper.RestartExplorer()</script>
</processing>
```

- **StartService (ServiceName, OptionalParam1, OptionalParam2,...).** Starts the service identified by *ServiceName*. *ServiceName* is the subkey in `HKLM\System\CurrentControlSet\Services` that holds the data for the given service. The optional parameters, if any, is passed to the StartService API. For more information, see the StartServiceA function (winsvc.h) article.

- **StopService (ServiceName)**. Stops the service that is identified by *ServiceName*. *ServiceName* is the subkey in `HKLM\System\CurrentControlSet\Services` that holds the data for the given service.
- **SyncSCM(ServiceShortName).** Reads the Start type value from the registry `(HKLM\System\CurrentControlSet\Services\ServiceShortName [Start])` after the value is changed by the migration engine, and then synchronizes Service Control Manager (SCM) with the new value.

<text>

The **<text>** element can be used to set a value for any environment variables that are inside one of the migration **.xml** files.

- **Number of occurrences:** Once in each <variable> element.
- **Parent elements:** <variable>
- **Child elements:** None.

Syntax:

```xml
<text>NormalText</text>
```

Expand table

Setting	Value
NormalText	This text is interpreted as normal text.

For example:

```xml
<variable name="QuickTime5or6DataSys">
  <text>%CSIDL_COMMON_APPDATA%\QuickTime</text>
</variable>
```

<unconditionalExclude>

The **<unconditionalExclude>** element excludes the specified files and registry values from the migration, regardless of the other include rules in any of the migration **.xml** files or in the `Config.xml` file. The objects declared here aren't migrated because this element takes precedence over all other rules. For example, even if there are explicit **<include>** rules to include `.mp3` files, if they're excluded with this option, then they aren't migrated.

Use this element to exclude all `.mp3` files from the source computer. Or, if backing up `C:\UserData` using another method, the entire folder can be excluded from the migration. Use this element with caution. If an application needs a file that is excluded, the application might not function properly on the destination computer.

- **Number of occurrences:** Unlimited.
- **Parent elements:** <rules>
- **Child elements:** <objectSet>

Syntax:

XML

```
<unconditionalExclude></unconditionalExclude>
```

The following **.xml** file excludes all `.mp3` files from migration. For additional examples of how to use this element, see the Exclude Files and Settings.

XML

```
<migration
urlid="http://www.microsoft.com/migration/1.0/migxmlext/excludefiles">
  <component context="System" type="Documents">
        <displayName>Test</displayName>
        <role role="Data">
            <rules>
             <unconditionalExclude>
                        <objectSet>
    <script>MigXmlHelper.GenerateDrivePatterns ("* [*.mp3]", "Fixed")
</script>
                        </objectSet>
             </unconditionalExclude>
            </rules>
        </role>
    </component>
</migration>
```

<variable>

The **<variable>** element is required in an **<environment>** element. For each **<variable>** element there must be one **<objectSet>**, **<script>**, or **<text>** element. The content of the **<variable>** element assigns a text value to the environment variable. This element has the following three options:

1. If the **<variable>** element contains a **<text>** element, then the value of the variable element is the value of the **<text>** element.

2. If the **<variable>** element contains a **<script>** element and the invocation of the script produces a non-null string, then the value of the **<variable>** element is the result of the script invocation.

3. If the **<variable>** element contains an **<objectSet>** element and the evaluation of the **<objectSet>** element produces at least one object pattern, then the value of the first object to match the resulting object pattern is the value of the variable element.

- **Number of occurrences:** Unlimited
- **Parent elements:** <environment>
- **Required child elements:** either <text>, or <script>, or <objectSet>

Syntax:

XML

```
<variable name="ID" remap=TRUE|FALSE>
</variable>
```

Expand table

Setting	Required?	Value
name	Yes	*ID* is a string value that is the name used to reference the environment variable. Microsoft recommends that *ID* start with the component's name to avoid namespace collisions. For example, if the component's name is MyComponent, and a variable is desired that is the component's install path, `MyComponent.InstallPath` could be specified.
remap	No, default = FALSE	Specifies whether to evaluate this environment variable as a remapping environment variable. Objects that are located in a path that is underneath this environment variable's value are automatically moved to where the environment variable points on the destination computer.

The following example is from the `MigApp.xml` file:

```XML
<environment>
   <variable name="HklmWowSoftware">
      <text>HKLM\Software</text>
   </variable>
   <variable name="WinZip8or9or10Exe">

<script>MigXmlHelper.GetStringContent("Registry","%HklmWowSoftware%\Microsoft\Windows\CurrentVersion\App Paths\winzip32.exe []")</script>
   </variable>
</environment>
```

<version>

The **<version>** element defines the version for the component, but doesn't affect the migration.

- **Number of occurrences:** zero or one
- **Parent elements:** <component>
- **Child elements:** none

Syntax:

```XML
<version>ComponentVersion</version>
```

Expand table

Setting	Required?	Value
ComponentVersion	Yes	The version of the component, which can contain patterns.

For example:

```XML
<version>4.*</version>
```

<windowsObjects>

The **<windowsObjects>** element is for USMT internal use only. Don't use this element.

Appendix

Specifying locations

- **Specifying encoded locations**. The encoded location used in all of the helper functions is an unambiguous string representation for the name of an object. The encoded location is composed of the node part, optionally followed by the leaf enclosed in square brackets. This format makes a clear distinction between nodes and leaves.

  For example, specify the file `C:\Windows\Notepad.exe` like this: `c:\Windows[Notepad.exe]`. Similarly, specify the directory `C:\Windows\System32` like this: `c:\Windows\System32`. (Notice the absence of the `[]` construct.)

  Representing the registry is similar. The default value of a registry key is represented as an empty `[]` construct. For example, the default value for the `HKLM\SOFTWARE\MyKey` registry key is `HKLM\SOFTWARE\MyKey[]`.

- **Specifying location patterns**. Specifying a location pattern is similar to specifying an actual location. The exception is that both the node and leaf part accept patterns. However, a pattern from the node doesn't extend to the leaf.

  For example, the pattern `c:\Windows\*` matches the Windows directory and all subdirectories, but it doesn't match any of the files in those directories. To match the files as well, `c:\Windows\*[*]` must be specified.

Internal USMT functions

The following functions are for internal USMT use only. Don't use them in an **.xml** file.

- *AntiAlias*
- *ConvertScreenSaver*
- *ConvertShowIEOnDesktop*
- *ConvertToOfficeLangID*

- *MigrateActiveDesktop*
- *MigrateAppearanceUPM*
- *MigrateDisplayCS*
- *MigrateDisplaySS*
- *MigrateIEAutoSearch*
- *MigrateMouseUPM*
- *MigrateSoundSysTray*
- *MigrateTaskBarSS*
- *SetPstPathInMapiStruc*

Valid version tags

The following version tags can be used with various helper functions:

- "CompanyName"
- "FileDescription"
- "FileVersion"
- "InternalName"
- "LegalCopyright"
- "OriginalFilename"
- "ProductName"
- "ProductVersion"

The following version tags contain values that can be compared:

- "FileVersion"
- "ProductVersion"

Related articles

- USMT XML reference.

Feedback

Was this page helpful? Yes No

Provide product feedback

Offline Migration Reference

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

Offline migration enables the **ScanState** tool to run inside a different Windows operating system than the Windows operating system from which **ScanState** is gathering files and settings. There are two primary offline scenarios:

- **Windows PE.** The **ScanState** tool can be run from within Windows PE, gathering files and settings from the offline Windows operating system on that machine.
- **Windows.old.** The **ScanState** tool can gather files and settings from the **Windows.old** directory. The **Windows.old** directory is created during Windows installation on a partition that contains a previous installation of Windows. For example, the **ScanState** tool can run in Windows, gathering files from a previous Windows installation contained in the **Windows.old** directory.

When using the User State Migration Tool (USMT) to gather and restore user state, offline migration reduces the cost of deployment by:

- **Reducing complexity.** In computer-refresh scenarios, migrations from the **Windows.old** directory reduce complexity by eliminating the need for the **ScanState** tool to be run before the operating system is deployed. Also, migrations from the **Windows.old** directory enable **ScanState** and **LoadState** to be run successively.
- **Improving performance.** When USMT runs in an offline Windows Preinstallation Environment (WinPE) environment, it has better access to the hardware resources. Running USMT in WinPE can increase performance on older machines with limited hardware resources and numerous installed software applications.
- **New recovery scenario.** In scenarios where a machine no longer restarts properly, it might be possible to gather user state with the **ScanState** tool from within WinPE.

What migrates offline?

The following user data and settings migrate offline, similar to an online migration:

- Data and registry keys specified in MigXML.
- User accounts.
- Application settings.

- Limited set of operating-system settings.
- EFS files.
- Favorites.

For exceptions to what can be migrated offline, see What Does USMT Migrate?

What offline environments are supported?

All currently supported

The following table defines the supported combination of online and offline operating systems in USMT.

Expand table

Running Operating System	Offline Operating System
Currently supported version of WinPE, with the MSXML library	Windows 7, Windows 8, Windows 10, Windows 11
Windows 10, Windows 11	**Windows.old** directory

> **Note**
>
> It is possible to run the **ScanState** tool while the drive remains encrypted by suspending Windows BitLocker Drive Encryption before booting into WinPE. For more information, see BitLocker operations guide: Suspend and resume. If using a Microsoft Configuration Manager task sequence, see Task sequence steps: Disable BitLocker.

User-group membership and profile control

User-group membership isn't preserved during offline migrations. A **<ProfileControl>** section must be configured in the `Config.xml` file to specify the groups that the migrated users should be made members of. The following example places all migrated users into the Users group:

```xml
<Configuration>
<ProfileControl>
```

```
    <localGroups>
      <mappings>
         <changeGroup from="*" to="Users" appliesTo="MigratedUsers">
            <include>
            <pattern>*</pattern>
            </include>
         </changeGroup>
      </mappings>
   </localGroups>
  </ProfileControl>
</Configuration>
```

For information about the format of a `Config.xml` file, see Config.xml File.

Command-line options

An offline migration can either be enabled by using a configuration file on the command line, or by using one of the following command line options:

Expand table

Component	Option	Description
ScanState.exe	**/offline:***<path to Offline.xml>*	This command-line option enables the offline-migration mode and requires a path to an Offline.xml configuration file.
ScanState.exe	**/offlineWinDir:** *<Windows directory>*	This command-line option enables the offline-migration mode and starts the migration from the location specified. It's only for use in WinPE offline scenarios where the migration is occurring from a Windows directory.
ScanState.exe	**/OfflineWinOld:** *<Windows.old directory>*	This command-line option enables the offline migration mode and starts the migration from the location specified. Only use in **Windows.old** migration scenarios, where the migration is occurring from a **Windows.old** directory.

Only one of the `/offline`, `/offlineWinDir`, or `/OfflineWinOld` command-line options can be used at a time. USMT doesn't support using more than one together.

Environment variables

System environment variables are necessary in the scenarios outlined in the following table:

Expand table

Variable	Value	Scenario
USMT_WORKING_DIR	Full path to a working directory	Required when USMT binaries are located on read-only media, which doesn't support the creation of log files or temporary storage. To set the system environment variable, at a command prompt type the following command: `Set USMT_WORKING_DIR=<path to working directory>`
MIG_OFFLINE_PLATFORM_ARCH	32 or 64	While operating offline, this environment variable defines the architecture of the offline system, if the system doesn't match the WinPE and `ScanState.exe` architecture. This environment variable enables the 32-bit **ScanState** application to gather data from a computer with 64-bit architecture, or the 64-bit **ScanState** application to gather data from a computer with 32-bit architecture. Specifying the architecture is required when auto-detection of the offline architecture doesn't function properly. For example, to set this system environment variable for a 32-bit architecture, at a command prompt type the following command: `Set MIG_OFFLINE_PLATFORM_ARCH=32`

Offline.xml elements

Use an `Offline.xml` file when running the **ScanState** tool on a computer that has multiple Windows directories. The `Offline.xml` file specifies which directories to scan for windows files. An `Offline.xml` file can be used with the `/offline` option as an alternative to specifying a single Windows directory path with the `/offlineDir` option.

<offline>

This element contains other elements that define how an offline migration is to be performed.

Syntax:

```
XML
```

```
<offline> </offline>
```

<winDir>

This element is a required child of **<offline>** and contains information about how the offline volume can be selected. The migration is performed from the first element of **<winDir>** that contains a valid Windows system volume.

Syntax:

```XML
<winDir> </winDir>
```

<path>

This element is a required child of **<winDir>** and contains a file path pointing to a valid Windows directory. Relative paths are interpreted from the **ScanState** tool's working directory.

Syntax:

```XML
<path> C:\Windows </path>
```

or when used with the **<mappings>** element:

Syntax:

```XML
<path> C:\, D:\ </path>
```

<mappings>

This element is an optional child of **<offline>**. When specified, the **<mappings>** element overrides the automatically detected WinPE drive mappings. Each child **<path>** element provides a mapping from one system volume to another. Additionally, mappings between folders can be provided, since an entire volume can be mounted to a specific folder.

Syntax:

```XML
<mappings> </mappings>
```

<failOnMultipleWinDir>

This element is an optional child of **<offline>**. The **<failOnMultipleWinDir>** element allows the user to specify that the migration should fail when USMT detects that there are multiple instances of Windows installed on the source machine. When the **<failOnMultipleWinDir>** element isn't present, the default behavior is that the migration doesn't fail.

Syntax:

```XML
<failOnMultipleWinDir>1</failOnMultipleWinDir>
```

or

Syntax:

```XML
<failOnMultipleWinDir>0</failOnMultipleWinDir>
```

Offline .xml Example

The following XML example illustrates some of the elements discussed earlier in this article.

```XML
<offline>
     <winDir>
          <path>C:\Windows</path>
          <path>D:\Windows</path>
          <path>E:\</path>
     </winDir>
     <failOnMultipleWinDir>1</failOnMultipleWinDir>
</offline>
```

Related articles

- Plan the migration.

Feedback

Was this page helpful? Yes No

Provide product feedback

User State Migration Tool (USMT) troubleshooting

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

The following table describes articles that address common User State Migration Tool (USMT) issues and questions. These articles describe tools that can be used to troubleshoot issues that arise during the migration.

In this section

Expand table

Link	Description
Common Issues	Find troubleshooting solutions for common problems in USMT.
Frequently Asked Questions	Find answers to questions about how to use USMT.
Log Files	Learn how to enable logging to help troubleshoot issues in USMT.
Return Codes	Learn how to use return codes to identify problems in USMT.
USMT Resources	Find more information and support for using USMT.

Related articles

- USMT best practices.
- User State Migration Tool (USMT) overview articles.
- User State Migration Tool (USMT) how-to articles.
- User State Migration Toolkit (USMT) reference.

Feedback

Was this page helpful? Yes No

Provide product feedback

User State Migration Tool (USMT) common issues

01/15/2025Applies to: Supported versions of Windows Client

The following sections discuss common issues that you might see when you run the USMT 10.0 tools. USMT produces log files that describe in further detail any errors that occurred during the migration process. These logs can be used to troubleshoot migration failures.

General guidelines for identifying migration problems

When you encounter a problem or error message during migration, you can use the following general guidelines to help determine the source of the problem:

- Examine the **ScanState**, **LoadState**, and UsmtUtils logs to obtain the exact USMT error messages and Windows® application programming interface (API) error messages. For more information about USMT return codes and error messages, see Return codes. You can obtain more information about any listed **Windows** system error codes by typing in a command prompt window `net.exe helpmsg <error_number>` where <*error_number*> is the error code number generated by the error message. For more information about System Error Codes, see System Error Codes (0-499).

  In most cases, the **ScanState** and **LoadState** logs indicate why a USMT migration is failing. We recommend that you use the `/v:5` option when testing your migration. This verbosity level can be adjusted in a production migration; however, reducing the verbosity level might make it more difficult to diagnose failures that are encountered during production migrations. You can use a verbosity level higher than 5 if you want the log files output to go to a debugger.

  > **Note**
  >
  > Running the **ScanState** and **LoadState** tools with the `/v:5` option creates a detailed log file. Although this option makes the log file large, the extra detail can help you determine where migration errors occurred.

- Use the `/Verify` option with the UsmtUtils tool to determine whether any files in a compressed migration store are corrupted. For more information, see Verify the condition of a compressed migration store.

- Use the `/Extract` option with the UsmtUtils tool to extract files from a compressed migration store. For more information, see Extract files from a compressed USMT migration store.
- Create a progress log using the `/Progress` option to monitor your migration.
- For the source and destination computers, obtain operating system information, and versions of applications such as Internet Explorer and any other relevant programs. Then verify the exact steps that are needed to reproduce the problem. This information might help you to understand what is wrong and to reproduce the issue in your testing environment.
- Sign out after you run the **LoadState** tool. Some settings such as fonts, desktop backgrounds, and screen-saver settings won't take effect until the next time the end user logs on.
- Close all applications before running **ScanState** or **LoadState** tools. If some applications are running during the **ScanState** or **LoadState** process, USMT might not migrate some data. For example, if Microsoft Outlook® is open, USMT might not migrate PST files.

> **Note**
>
> USMT will fail if it can't migrate a file or setting unless you specify the `/c` option. When you specify the `/c` option, USMT ignores errors. However, it logs an error when it encounters a file that is in use that didn't migrate.

User account problems

The following sections describe common user account problems. Expand the section to see recommended solutions.

I'm having problems creating local accounts on the destination computer

Resolution: For more information about creating accounts and migrating local accounts, see Migrate user accounts.

Not all of the user accounts were migrated to the destination computer

Causes/Resolutions There are two possible causes for this problem:

When running the **ScanState** and LoadState tools on Windows 7, Windows 8, or Windows 10, you must run them in Administrator mode from an account with administrative credentials to ensure that all specified users are migrated. To run in Administrator mode:

1. Select **Start** > **All Programs** > **Accessories**.
2. Right-click **Command Prompt**.
3. Select **Run as administrator**.
4. Specify the *LoadState.exe* or *ScanState.exe* command.

If you don't run USMT in Administrator mode, only the user profile that is logged on will be included in the migration.

Any user accounts on the computer that haven't been used won't be migrated. For example, if you add User1 to the computer, but User1 never logs on, then USMT won't migrate the User1 account.

User accounts that I excluded were migrated to the destination computer

Cause: The command that you specified might have had conflicting `ui` and `/ue` options. If a user is specified with the `/ui` option and with either the `/ue` or `/uel` options at the same time, the user will be included in the migration. For example, if you specify `/ui:domain1\*` `/ue:domain1\user1`, then User1 will be migrated because the `/ui` option takes precedence.

Resolution: For more information about how to use the `/ui` and `/ue` options together, see the examples in the ScanState Syntax article.

I'm using the /uel option, but many accounts are still being included in the migration

Cause: The `/uel` option depends on the last modified date of the users' NTUser.dat file. There are scenarios in which this last modified date might not match the users' last sign-in date.

Resolution: This is a limitation of the `/uel` option. You might need to exclude these users manually with the `/ue` option.

The LoadState tool reports an error as return code 71 and fails to restore a user profile during a migration test

Cause: During a migration test, if you run the **ScanState** tool on your test computer and then delete user profiles in order to test the **LoadState** tool on the same computer, you may have a conflicting key present in the registry. Using the **net use** command to remove a user profile will delete folders and files associated with that profile, but won't remove the registry key.

Resolution: To delete a user profile, use the **User Accounts** item in Control Panel. To correct an incomplete deletion of a user profile:

1. Open the registry editor by typing *regedit.exe* at an elevated command prompt.
2. Navigate to `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\ProfileList`.

   Each user profile is stored in a System Identifier key under `ProfileList`.
3. Delete the key for the user profile you're trying to remove.

Files that weren't encrypted before the migration are now encrypted with the account used to run the LoadState tool

Cause: The **ScanState** tool was run using the `/EFS:copyraw` option to migrate encrypted files and Encrypting File System (EFS) certificates. The encryption attribute was set on a folder that was migrated, but the attribute was removed from file contents of that folder prior to migration.

Resolution: Before using the **ScanState** tool for a migration that includes encrypted files and EFS certificates, you can run the Cipher tool at the command prompt to review and change encryption settings on files and folders. You must remove the encryption attribute from folders that contain unencrypted files or encrypt the contents of all files within an encrypted folder.

To remove encryption from files that have already been migrated incorrectly, you must sign into the computer with the account that you used to run the **LoadState** tool and then remove the encryption from the affected files.

The LoadState tool reports an error as return code 71 and a Windows Error 2202 in the log file

Cause: The computer name was changed during an offline migration of a local user profile.

Resolution: You can use the `/mu` option when you run the **LoadState** tool to specify a new name for the user. For example,

Windows Command Prompt

```
LoadState.exe  /i:MigApp.xml /i:MigDocs.xml \\server\share\migration\mystore
/progress:Progress.log /l:LoadState.log /mu:fareast\user1:farwest\user1
```

Command-line problems

The following sections describe common command-line problems. Expand the section to see recommended solutions.

I received the following error message: "Usage Error: You can't specify a file path with any of the command-line options that exceeds 256 characters."

Cause: You might receive this error message in some cases even if you don't specify a long store or file path, because the path length is calculated based on the absolute path. For example, if you run the `**ScanState**.exe /o store` command from *C:\Program Files\USMT40*, then each character in *C:\Program Files\USMT40* will be added to the length of "store" to get the length of the path.

Resolution: Ensure that the total path length doesn't exceed 256 characters. The total path length includes the store path plus the current directory.

I received the following error message: "USMT was unable to create the log file(s). Ensure that you have write access to the log directory."

Cause: If you're running the **ScanState** or **LoadState** tools from a shared network resource, you'll receive this error message if you don't specify `/l`.

Resolution: To fix this issue in this scenario, specify the `/l:ScanState.log` or `/l:LoadState.log` option.

XML file problems

The following sections describe common XML file problems. Expand the section to see recommended solutions.

I used the `/genconfig` option to create a Config.xml file, but I see only a few applications and components that are in MigApp.xml. Why does Config.xml not contain all of the same applications?

Cause: *Config.xml* will contain only operating system components, applications, and the user document sections that are in both of the .xml files and are installed on the computer when you run the `/genconfig` option. Otherwise, these applications and components won't appear in the *Config.xml* file.

Resolution: Install all of the desired applications on the computer before running the `/genconfig` option. Then run *ScanState.exe* with all of the .xml files. For example, run the following command:

Windows Command Prompt

```
ScanState.exe /genconfig:Config.xml /i:MigDocs.xml /i:MigApp.xml /v:5
/l:ScanState.log
```

I'm having problems with a custom .xml file that I authored, and I can't verify that the syntax is correct

Resolution: You can load the XML schema file *MigXML.xsd* into your XML authoring tool. *MigXML.xsd* is included with USMT. For examples, see the Visual Studio Development Center. Then, load your .xml file in the authoring tool to see if there's a syntax error. For more information about using the XML elements, see USMT XML Reference.

I'm using a MigXML helper function, but the migration isn't working the way I expected it to. How do I troubleshoot this issue?

Cause: Typically, this issue is caused by incorrect syntax used in a helper function. You receive a Success return code, but the files you wanted to migrate didn't get collected or applied, or weren't collected or applied in the way you expected.

Resolution: You should search the **ScanState** or **LoadState** log for either the component name that contains the MigXML helper function, or the MigXML helper function title, so that you can locate the related warning in the log file.

Migration problems

The following sections describe common migration problems. Expand the section to see recommended solutions.

Files that I specified to exclude are still being migrated

Cause: There might be another rule that is including the files. If there's a more specific rule or a conflicting rule, the files will be included in the migration.

Resolution: For more information, see Conflicts and Precedence and the Diagnostic Log section in Log Files.

I specified rules to move a folder to a specific location on the destination computer, but it hasn't migrated correctly

Cause: There might be an error in the XML syntax.

Resolution: You can use the USMT XML schema (*MigXML.xsd*) to write and validate migration .xml files. Also see the XML examples in the following articles:

Conflicts and precedence

Exclude files and settings

Reroute files and settings

Include files and settings

Custom XML examples

After LoadState completes, the new desktop background doesn't appear on the destination computer

There are three typical causes for this issue.

Cause: Some settings such as fonts, desktop backgrounds, and screen-saver settings aren't applied by **LoadState** until after the destination computer has been restarted.

Resolution: To fix this issue, sign out, and then log back on to see the migrated desktop background.

I included MigApp.xml in the migration, but some PST files aren't migrating

Cause: The *MigApp.xml* file migrates only the PST files that are linked to Outlook profiles.

Resolution: To migrate PST files that aren't linked to Outlook profiles, you must create a separate migration rule to capture these files.

USMT doesn't migrate the Start layout

Description: You're using USMT to migrate profiles from one installation of Windows 10 to another installation of Windows 10 on different hardware. After migration, the user signs in on the new device and doesn't have the Start menu layout they had previously configured.

Cause: A code change in the Start Menu with Windows 10 version 1607 and later is incompatible with this USMT function.

Resolution: The following workaround is available:

1. With the user signed in, back up the Start layout using the following Windows PowerShell command. You can specify a different path if desired:

   PowerShell
   ```
   Export-StartLayout -Path "C:\Layout\user1.xml"
   ```

2. Migrate the user's profile with USMT.

3. Before the user signs in on the new device, import the Start layout using the following Windows PowerShell command:

   PowerShell
   ```
   Import-StartLayout -LayoutPath "C:\Layout\user1.xml" -MountPath %systemdrive%
   ```

This workaround changes the Default user's Start layout. The workaround doesn't scale to a mass migrations or multiuser devices, but it can potentially unblock some scenarios. If other users will sign on to the device, you should delete layoutmodification.xml from the Default user profile. Otherwise, all users who sign on to that device will use the imported Start layout.

Offline migration problems

The following sections describe common offline migration problems. Expand the section to see recommended solutions.

Some of my system settings don't migrate in an offline migration

Cause: Some system settings, such as desktop backgrounds and network printers, aren't supported in an offline migration. For more information, see What does USMT migrate?

Resolution: In an offline migration, these system settings must be restored manually.

The ScanState tool fails with return code 26

Cause: A common cause of return code 26 is that a temp profile is active on the source computer. This profile maps to c:\users\temp. The **ScanState** log shows a **MigStartupOfflineCaught** exception that includes the message **User profile duplicate SID error**.

Resolution: You can reboot the computer to get rid of the temp profile or you can set **MIG_FAIL_ON_PROFILE_ERROR=0** to skip the error and exclude the temp profile.

Include and Exclude rules for migrating user profiles don't work the same offline as they do online

Cause: When offline, the DNS server can't be queried to resolve the user name and SID mapping.

Resolution: Use a Security Identifier (SID) to include a user when running the **ScanState** tool. For example:

Windows Command Prompt

```
ScanState.exe /ui:S1-5-21-124525095-708259637-1543119021*
```

The wild card (*) at the end of the SID will migrate the *SID*_Classes key as well.

You can also use patterns for SIDs that identify generic users or groups. For example, you can use the `/ue:*-500` option to exclude the local administrator accounts. For more information about Windows SIDs, see Security identifiers.

My script to wipe the disk fails after running the ScanState tool on a 64-bit system

Cause: The HKLM registry hive isn't unloaded after the **ScanState** tool has finished running.

Resolution: Reboot the computer or unload the registry hive at the command prompt after the **ScanState** tool has finished running. For example, at a command prompt, enter:

Windows Command Prompt

```
reg.exe unload hklm\$dest$software
```

Hard-Link Migration Problems

The following sections describe common hard-link migration problems. Expand the section to see recommended solutions.

EFS files aren't restored to the new partition

Cause: EFS files can't be moved to a new partition with a hard link. The `/efs:hardlink` command-line option is only applicable to files migrated on the same partition.

Resolution: Use the `/efs:copyraw` command-line option to copy EFS files during the migration instead of creating hard links, or manually copy the EFS files from the hard-link store.

The ScanState tool can't delete a previous hard-link migration store

Cause: The migration store contains hard links to locked files.

Resolution: Use the UsmtUtils tool to delete the store or change the store name. For example, at a command prompt, enter:

Windows Command Prompt

```
UsmtUtils.exe /rd <storedir>
```

You should also reboot the machine.

Data collection

If you need assistance from Microsoft support, we recommend you collect the information by following the steps mentioned in Gather information by using TSS for deployment-related issues.

Related articles

User State Migration Tool (USMT) troubleshooting

Frequently asked questions

Return codes

UsmtUtils syntax

Frequently Asked Questions

Applies to:

- Windows 11
- Windows 10

The following sections provide frequently asked questions and recommended solutions for migrations using User State Migration Tool (USMT).

General

How much space is needed on the destination computer?

The destination computer needs enough available space for the following items:

- Operating system
- Applications
- Uncompressed store

Can the files and settings be stored directly on the destination computer or is a server needed?

Files don't need to be saved to a server. If moving the user state to a new computer, the store can be created on:

- A shared folder.
- On removable media, such as a USB flash drive (UFD).
- Directly on the destination computer.

To store it directly on the destination computer:

1. Create and share the directory `C:\store` on the destination computer.
2. Run the **ScanState** tool on the source computer and save the files and settings to `\\<DestinationComputerName>\store`
3. Run the **LoadState** tool on the destination computer and specify `C:\store` as the store location.

Can data be migrated between operating systems with different languages?

No. USMT doesn't support migrating data between operating systems with different languages; the source computer's operating-system language must match the destination computer's operating-system language.

Can the location of the temporary directory on the destination computer be changed?

Yes. The environment variable `USMT\_WORKING\_DIR` can be changed to an alternative temporary directory. There are some offline migration scenarios where changing the temporary directory is necessary, for example, when the USMT binaries are located on read-only Windows Preinstallation Environment (WinPE) boot media.

How is USMT installed?

Because USMT is included in Windows Assessment and Deployment Kit (Windows ADK), the Windows ADK package needs to be installed on at least one computer in the environment. The USMT binaries can then be copied from the USMT directory located on the original computer where the Windows ADK was installed to additional client computers.

How is USMT uninstalled?

For computers that have the Windows ADK installed, uninstalling the Windows ADK from the computer uninstalls USMT. For client computers that don't have the Windows ADK installed, the USMT directory can be deleted to uninstall USMT.

Files and Settings

How can a folder or a certain type of file be excluded from the migration?

The <**unconditionalExclude**> element can be used to globally exclude data from the migration. For example, this element can be used to exclude all MP3 files on the computer or to exclude all files from `C:\UserData`. This element excludes objects regardless of any other <**include**> rules that are in the **.xml** files. For an example, see <**unconditionalExclude**> in the Exclude files and settings article. For the syntax of this element, see XML elements library.

What happens to files that were located on a drive that don't exist on the destination computer?

USMT migrates the files to the `%SystemDrive%` while maintaining the correct folder hierarchy. For example:

- `E:\data\File.pst` is on the source computer.
- Destination computer doesn't have an E:\ drive.
- C:\ is the system drive on the destination computer.

the file is migrated to `C:\data\File.pst`. This behavior holds true even when <**locationModify**> rules attempt to move data to a drive that doesn't exist on the destination computer.

USMT .xml Files

Where are there examples of USMT **.xml** files?

The following articles include examples of USMT **.xml** files:

- Exclude files and settings
- Reroute files and settings
- Include files and settings
- Custom XML examples

Can custom **.xml** files that were written for USMT 5.0 be used?

Yes. Custom **.xml** files that were written for USMT 5.0 can be used with newer versions of USMT. However, in order to use new USMT functionality, the custom USMT files must be revisited and refreshed to include the new command-line options and XML elements.

How can the **.xml** files be validated?

The USMT XML Schema (`MigXML.xsd`) can be used to write and validate migration **.xml** files.

Why must the **.xml** files be included with both the `ScanState.exe` and `LoadState.exe` commands?

The **.xml** files aren't copied to the store as in previous versions of USMT. Because the **ScanState** and **LoadState** tools need the **.xml** files to control the migration, the same set of **.xml** files must be specified for the `ScanState.exe` and `LoadState.exe` commands. If a particular set of mig*.xml files were used in the **ScanState** tool, either called through the `/auto` option, or individually through the `/i` option, then the same option should be used to call the exact same mig*.xml files in the **LoadState** tool. However, the `Config.xml` file doesn't need to be specified, unless files and settings that were migrated to the store need to be excluded. For example, the **Documents** folder might be migrated to the store, but not to the destination computer. To do this type of migration, modify the `Config.xml` file and specify the updated file with the `LoadState.exe` command. **LoadState** migrates only the desired files and settings.

If an **.xml** file is excluded from the `LoadState.exe` command, then all of the data in the store that was migrated with the missing **.xml** files are migrated. However, the migration rules that were specified for the `ScanState.exe` command don't apply. For example, if a `MigApp.xml` file that has a rerouting rule such as `MigsysHelperFunction.RelativeMove("c:\data", "%CSIDL_PERSONAL%")` is excluded, USMT doesn't reroute the files. Instead, it migrates them to `C:\data`.

Which files can be modified and specified on the command line?

The `MigUser.xml`, `MigApp.xml`, and `MigDocs.xml` files can be specified on the command line. Each of these files can be modified. Manifests control the migration of operating system settings. Manifests can't be modified. To exclude certain operating-system settings or any other components, create and modify the `Config.xml` file.

What happens if the **.xml** files aren't specified on the command line?

- **ScanState**

  If no files are specified with the `ScanState.exe` command, all user accounts and default operating system components are migrated.

- **LoadState**

If no files are specified with the `LoadState.exe` command, all data that is in the store is migrated. However, any target-specific migration rules that were specified in **.xml** files with the `ScanState.exe` command doesn't apply. For example, if a `MigApp.xml` file that has a rerouting rule such as `MigsysHelperFunction.RelativeMove("c:\data", "%CSIDL_PERSONAL%")` is excluded, USMT doesn't reroute the files. Instead, it migrates them to `C:\data`.

Conflicts and Precedence

What happens when there are conflicting XML rules or conflicting objects on the destination computer?

For more information, see Conflicts and precedence.

Related topics

- User State Migration Tool (USMT) Troubleshooting.
- Extract files from a compressed USMT migration store.
- Verify the condition of a compressed migration store.

USMT log files

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

User State Migration Tool (USMT) logs can be used to monitor the migration and to troubleshoot errors and failed migrations. This article describes the available command-line options to enable USMT logs. It also describes new XML elements that can be used to configure:

- Which types of errors are fatal and should halt the migration.
- Which types are non-fatal and should be skipped so that the migration can continue.

Log command-line options

The following table describes each command-line option related to logs, and it provides the log name and a description of what type of information each log contains.

Expand table

Command line Option	File Name	Description
/l"*[Path]FileName*	`ScanState.exe.log` or `LoadState.log`	Specifies the path and file name of the **ScanState** log or **LoadState** log.
/progress:*[Path]FileName*	Specifies the path and file name of the Progress log.	Provides information about the status of the migration, by percentage complete.
/v:*[VerbosityLevel]*	Not applicable	See Monitoring options in ScanState syntax.
/listfiles:*[Path]FileName*	Specifies the path and file name of the Listfiles log.	Provides a list of the files that were migrated.
Set the environment variable **MIG_ENABLE_DIAG** to a path to an XML file.	`USMTDiag.xml`	The diagnostic log contains detailed system environment information, user environment information, and information about the migration units (migunits) being gathered and their contents.

ⓘ Note

The log files can't be stored in *StorePath*. If the log files are stored in *StorePath*, the log files are overwritten when USMT runs.

ScanState and LoadState logs

ScanState and **LoadState** logs are text files that are created when the **ScanState** and **LoadState** tools run. These logs can be used to help monitor the migration. The content of the log depends on the command-line options that are used and the verbosity level that is specified. For more information about verbosity levels, see Monitoring options in ScanState syntax.

Progress log

A progress log can be created using the `/progress` option. External tools, such as Microsoft System Center Operations Manager, can parse the progress log to update the monitoring systems. The first three fields in each line are fixed as follows:

- **Date:** Date, in the format of *day shortNameOfTheMonth year*. For example: 08 Jun 2023.
- **Local time:** Time, in the format of *hrs:minutes:seconds* (using a 24-hour clock). For example: 13:49:13.
- **Migration time:** Duration of time that USMT was run, in the format of *hrs:minutes:seconds*. For example: 00:00:20.

The remaining fields are key/value pairs as indicated in the following table.

Expand table

Key	Value
program	`ScanState.exe` or `LoadState.exe`.
productVersion	The full product version number of USMT.
computerName	The name of the source or destination computer on which USMT was run.
commandLine	The full command used to run USMT.
PHASE	Reports that a new phase in the migration is starting. This key can be one of the following values: • Initializing

Key	Value
	• Scanning • Collecting • Saving • Estimating • Applying
detectedUser	• For the **ScanState** tool, this key is the users USMT detected on the source computer that can be migrated. • For the **LoadState** tool, this key is the users USMT detected in the store that can be migrated.
includedInMigration	Defines whether the user profile/component is included for migration. Valid values are **Yes** or **No**.
forUser	Specifies either of the following values: • The user state being migrated. • *This Computer*, meaning files and settings that aren't associated with a user.
detectedComponent	Specifies a component detected by USMT. • For *ScanState*, this key is a component or application that is installed on the source computer. • For **LoadState**, this key is a component or application that was detected in the store.
totalSizeInMBToTransfer	Total size of the files and settings to migrate in megabytes (MB).
totalPercentageCompleted	Total percentage of the migration that is completed by either **ScanState** or **LoadState**.
collectingUser	Specifies which user **ScanState** is collecting files and settings for.
totalMinutesRemaining	Time estimate, in minutes, for the migration to complete.
error	Type of non-fatal error that occurred. This key can be one of the following values: • **UnableToCopy**: Unable to copy to store because the disk on which the store is located is full. • **UnableToOpen**: Unable to open the file for migration because another application or service has the file open in non-shared mode. • **UnableToCopyCatalog**: Unable to copy because the store is corrupted. • **UnableToAccessDevice**: Unable to access the device.

Key	Value
	• **UnableToApply**: Unable to apply the setting to the destination computer.
objectName	The name of the file or setting that caused the non-fatal error.
action	Action taken by USMT for the non-fatal error. The values are: • **Ignore**: Non-fatal error ignored and the migration continued because the **/c** option was specified on the command line. • **Abort**: Stopped the migration because the **/c** option wasn't specified.
errorCode	The errorCode or return value.
numberOfIgnoredErrors	The total number of non-fatal errors that USMT ignored.
*message**	The message corresponding to the errorCode.

List files log

The List files log (`Listfiles.txt`) provides a list of the files that were migrated. This list can be used to troubleshoot XML issues or can be retained as a record of the files that were gathered into the migration store. The List Files log is only available for `ScanState.exe`.

Diagnostic log

The diagnostic log can be obtained by setting the environment variable **MIG_ENABLE_DIAG** to a path to an XML file.

The diagnostic log contains:

- Detailed system environment information.
- Detailed user environment information.
- Information about the migration units (migunits) being gathered and their contents.

Using the Diagnostic Log

The diagnostic log is essentially a report of all the migration units (migunits) included in the migration. A migunit is a collection of data. In the XML files, the component identifies the migunit that the migunit is associated with. The migration store is made up of all the migunits in the migration. The diagnostic log can be used to verify which migunits were included in the migration and can be used for troubleshooting while authoring migration XML files.

The following examples describe common scenarios in which the diagnostic log can be used.

Why is this file not migrating when I authored an "include" rule for it?

Let's imagine that we have the following directory structure and that we want the **data** directory to be included in the migration along with the **New Text Document.txt** file in the **New Folder**. The directory of `C:\data` contains:

Windows Command Prompt

```
12/21/2023  01:08 PM    <DIR>          .
12/21/2023  01:08 PM    <DIR>          ..
12/21/2023  01:08 PM    <DIR>          New Folder
12/21/2023  01:19 PM                13 test (1).txt
12/21/2023  01:19 PM                13 test.txt
               2 File(s)             26 bytes
```

The directory of `C:\data\New Folder` contains:

Windows Command Prompt

```
12/21/2023  01:08 PM    <DIR>          .
12/21/2023  01:08 PM    <DIR>          ..
12/21/2023  01:08 PM                 0 New Text Document.txt
               1 File(s)              0 bytes
```

To migrate these files the following migration XML is authored:

XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<migration urlid="http://www.microsoft.com/migration/1.0/TestSuite_BUGFIX">

<component context="System"  type="Application">
  <displayName>DATA1</displayName>
  <role role="Data">
    <rules>
      <include>
        <objectSet>
```

```
          <pattern type="File">c:\data\ [*]</pattern>
        </objectSet>
      </include>

    </rules>
  </role>
</component>
</migration>
```

However, upon testing the migration, the **New Text Document.txt** file is noticed that it wasn't included in the migration. To troubleshoot this failure, the migration can be repeated with the environment variable **MIG_ENABLE_DIAG** set such that the diagnostic log is generated. Searching the diagnostic log for the component **DATA1** reveals the following XML section:

XML

```
<MigUnitList>
  <MigUnit Name="\<System\>\DATA1 (CMXEAgent)" Context="System"
ConfidenceLevel="100" Group="Applications" Role="UserData" Agent="CMXEAgent"
Selected="true" Supported="true">
    <Patterns Type="Include">
      <Pattern Type="File" Path="C:\data [*]"/>
    </Patterns>
  </MigUnit>
</MigUnitList>
<Perform Name="Gather" User="System">
  <MigUnit Name="\<System\>\DATA1 (CMXEAgent)">
    <Operation Name="Store" Type="File" Path="C:\data" SimObj="false"
Success="true"/>
    <Operation Name="Store" Type="File" Path="C:\data [test (1).txt]"
SimObj="false" Success="true"/>
    <Operation Name="Store" Type="File" Path="C:\data [test.txt]"
SimObj="false" Success="true"/>
  </MigUnit>
</Perform>
```

Analysis of this XML section reveals the migunit that was created when the migration rule was processed. The **<Perform>** section details the actual files that were scheduled for gathering and the result of the gathering operation. The **New Text Document.txt** file doesn't appear in this section, which confirms that the migration rule wasn't correctly authored.

An analysis of the XML elements library reference article reveals that the **<pattern>** tag needs to be modified as follows:

XML

```
<pattern type="File">c:\data\* [*]</pattern>
```

When the migration is performed again with the modified tag, the diagnostic log reveals the following information:

XML

```
<MigUnitList>
  <MigUnit Name="\<System\>\DATA1 (CMXEAgent)" Context="System"
ConfidenceLevel="100" Group="Applications" Role="UserData" Agent="CMXEAgent"
Selected="true" Supported="true">
    <Patterns Type="Include">
      <Pattern Type="File" Path="C:\data\* [*]"/>
    </Patterns>
  </MigUnit>
</MigUnitList>
<Perform Name="Gather" User="System">
  <MigUnit Name="\<System\>\DATA1 (CMXEAgent)">
    <Operation Name="Store" Type="File" Path="C:\data" SimObj="false"
Success="true"/>
    <Operation Name="Store" Type="File" Path="C:\data [test (1).txt]"
SimObj="false" Success="true"/>
    <Operation Name="Store" Type="File" Path="C:\data [test.txt]"
SimObj="false" Success="true"/>
    <Operation Name="Store" Type="File" Path="C:\data\New Folder"
SimObj="false" Success="true"/>
    <Operation Name="Store" Type="File" Path="C:\data\New Folder [New Text
Document.txt]" SimObj="false" Success="true"/>
  </MigUnit>
</Perform>
```

This diagnostic log confirms that the modified **<pattern>** value enables the migration of the file.

Why is this file migrating when I authored an exclude rule excluding it?

In this scenario, the following directory structure exists and all files in the **Data** directory should migrate, except for text files. The `C:\Data` folder contains:

Windows Command Prompt

```
Directory of C:\Data

12/21/2023  01:08 PM    <DIR>          .
12/21/2023  01:08 PM    <DIR>          ..
12/21/2023  01:08 PM    <DIR>          New Folder
12/21/2023  01:19 PM                13 test (1).txt
```

```
12/21/2023  01:19 PM                13 test.txt
               2 File(s)             26 bytes
```

The `C:\Data\New Folder\` contains:

Windows Command Prompt

```
12/21/2023  01:08 PM    <DIR>          .
12/21/2023  01:08 PM    <DIR>          ..
12/21/2023  01:08 PM                 0 New Text Document.txt
               1 File(s)              0 bytes
```

The following migration XML is authored:

XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<migration urlid="http://www.microsoft.com/migration/1.0/TestSuite_BUGFIX">

<component context="System"  type="Application">
  <displayName>DATA1</displayName>
  <role role="Data">
    <rules>
      <include>
        <objectSet>
          <pattern type="File">c:\data\* [*]</pattern>
        </objectSet>
      </include>
    </rules>
    <rules>
      <exclude>
        <objectSet>
          <pattern type="File">c:\* [*.txt]</pattern>
        </objectSet>
      </exclude>

    </rules>
  </role>
</component>
```

However, upon testing the migration, all the text files are noticed that they're still included in the migration. In order to troubleshoot this issue, the migration can be performed with the environment variable **MIG_ENABLE_DIAG** set so that the diagnostic log is generated. Searching the diagnostic log for the component **DATA1** reveals the following XML section:

XML

```
<MigUnitList>
  <MigUnit Name="\<System\>\DATA1 (CMXEAgent)" Context="System"
ConfidenceLevel="100" Group="Applications" Role="UserData" Agent="CMXEAgent"
Selected="true" Supported="true">
    <Patterns Type="Include">
      <Pattern Type="File" Path="C:\data\* [*]"/>
    </Patterns>
    <Patterns Type="Exclude">
      <Pattern Type="File" Path="C:\* [*.txt]"/>
    </Patterns>
  </MigUnit>
</MigUnitList>
<Perform Name="Gather" User="System">
  <MigUnit Name="\<System\>\DATA1 (CMXEAgent)">
    <Operation Name="Store" Type="File" Path="C:\data" SimObj="false"
Success="true"/>
    <Operation Name="Store" Type="File" Path="C:\data [test (1).txt]"
SimObj="false" Success="true"/>
    <Operation Name="Store" Type="File" Path="C:\data [test.docx]"
SimObj="false" Success="true"/>
    <Operation Name="Store" Type="File" Path="C:\data [test.txt]"
SimObj="false" Success="true"/>
    <Operation Name="Store" Type="File" Path="C:\data\New Folder"
SimObj="false" Success="true"/>
    <Operation Name="Store" Type="File" Path="C:\data\New Folder [New Text
Document.txt]" SimObj="false" Success="true"/>
    <Operation Name="Store" Type="File" Path="C:\data\New Folder
[test.docx]" SimObj="false" Success="true"/>
  </MigUnit>
</Perform>
```

When the diagnostic log is reviewed, the files are still migrating is confirmed, and that it's a problem with the authored migration XML rule. An update is authored to the migration XML script as follows:

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<migration urlid="http://www.microsoft.com/migration/1.0/TestSuite_BUGFIX">

<component context="System"  type="Application">
  <displayName>DATA1</displayName>
  <role role="Data">
    <rules>
      <include>
        <objectSet>
          <pattern type="File">c:\data\* [*]</pattern>
        </objectSet>
      </include>
    </rules>
    <rules>
      <exclude>
```

```
        <objectSet>
          <pattern type="File">c:\data\* [*.txt]</pattern>
        </objectSet>
      </exclude>

    </rules>
  </role>
</component>


</migration>
```

The revised migration XML script excludes the files from migrating, as confirmed in the diagnostic log:

XML

```
<MigUnitList>
  <MigUnit Name="\<System\>\DATA1 (CMXEAgent)" Context="System"
ConfidenceLevel="100" Group="Applications" Role="UserData" Agent="CMXEAgent"
Selected="true" Supported="true">
    <Patterns Type="Include">
      <Pattern Type="File" Path="C:\data\* [*]"/>
    </Patterns>
    <Patterns Type="Exclude">
      <Pattern Type="File" Path="C:\data\* [*.txt]"/>
    </Patterns>
  </MigUnit>
</MigUnitList>
<Perform Name="Gather" User="System">
  <MigUnit Name="\<System\>\DATA1 (CMXEAgent)">
    <Operation Name="Store" Type="File" Path="C:\data" SimObj="false"
Success="true"/>
    <Operation Name="Store" Type="File" Path="C:\data [test.docx]"
SimObj="false" Success="true"/>
    <Operation Name="Store" Type="File" Path="C:\data\New Folder"
SimObj="false" Success="true"/>
    <Operation Name="Store" Type="File" Path="C:\data\New Folder
[test.docx]" SimObj="false" Success="true"/>
  </MigUnit>
</Perform>
```

Related articles

- XML elements library.
- ScanState syntax.
- LoadState syntax.

Feedback

Was this page helpful? Yes No

Provide product feedback

User State Migration Tool (USMT) return codes

01/15/2025Applies to: Supported versions of Windows Client

This article describes USMT 10.0 return codes and error messages. Also included is a table listing the USMT return codes with their associated mitigation steps. In addition, this article provides tips to help you use the logfiles to determine why you received an error.

Understanding the requirements for running USMT can help minimize errors in your USMT migrations. For more information, see USMT Requirements.

Return codes used by USMT

If you encounter an error in your USMT migration, you can use return codes and the more specific information provided in the associated USMT error messages to troubleshoot the issue and to identify mitigation steps.

Return codes are grouped into the following broad categories that describe their area of error reporting:

- Success or User Cancel
- Invalid Command Lines
- Setup and Initialization
- Non-fatal Errors
- Fatal Errors

As a best practice, we recommend that you set verbosity level to 5, `v:5`, on the *ScanState.exe*, *LoadState.exe*, and *UsmtUtils.exe* command lines so that the most detailed reporting is available in the respective USMT logs. You can use a higher verbosity level if you want the log files output to go to a debugger.

USMT error messages

Error messages provide more detailed information about the migration problem than the associated return code. For example, the **ScanState**, **LoadState**, or **UsmtUtils** tool might return a code of **11** (for **USMT_INVALID_PARAMETERS**) and a related error message that reads **/key and /keyfile both specified**. The error message is displayed at the command prompt and is

identified in the **ScanState**, **LoadState**, or **UsmtUtils** log files to help you determine why the return code was received.

You can obtain more information about any listed **Windows** system error codes by typing in a command prompt window `net.exe helpmsg <error_number>` where <*error_number*> is the error code number generated by the error message. For more information about System Error Codes, see System Error Codes (0-499).

Troubleshooting return codes and error messages

The following information lists each return code by numeric value, along with the associated error messages and suggested troubleshooting actions.

0: USMT_SUCCESS

- **Category**: Success or User Cancel

Expand table

Error message	Troubleshooting, mitigation, workarounds
Successful run	NA

1: USMT_DISPLAY_HELP

- **Category**: Success or User Cancel

Expand table

Error message	Troubleshooting, mitigation, workarounds
Command line help requested	NA

2: USMT_STATUS_CANCELED

- **Category**: Success or User Cancel

Expand table

Error message	Troubleshooting, mitigation, workarounds
Gather was aborted because of an EFS file	NA

Error message	Troubleshooting, mitigation, workarounds
User chose to cancel (such as pressing CTRL+C)	NA

3: USMT_WOULD_HAVE_FAILED

- Category:

Expand table

Error message	Troubleshooting, mitigation, workarounds
At least one error was skipped as a result of /c.	Review ScanState, LoadState, or UsmtUtils log for details about command-line errors.

11: USMT_INVALID_PARAMETERS

- **Category**: Invalid Command Lines

Expand table

Error message	Troubleshooting, mitigation, workarounds
/all conflicts with /ui, /ue or /uel	Review ScanState log or LoadState log for details about command-line errors.
/auto expects an optional parameter for the script folder	Review ScanState log or LoadState log for details about command-line errors.
/encrypt can't be used with /nocompress	Review ScanState log or LoadState log for details about command-line errors.
/encrypt requires /key or /keyfile	Review ScanState log or LoadState log for details about command-line errors.
/genconfig can't be used with most other options	Review ScanState log or LoadState log for details about command-line errors.
/genmigxml can't be used with most other options	Review ScanState log or LoadState log for details about command-line errors.
/hardlink requires /nocompress	Review ScanState log or LoadState log for details about command-line errors.
/key and /keyfile both specified	Review ScanState log or LoadState log for details about command-line errors.

Error message	Troubleshooting, mitigation, workarounds
/key or /keyfile used without enabling encryption	Review ScanState log or LoadState log for details about command-line errors.
/lae is only used with /lac	Review ScanState log or LoadState log for details about command-line errors.
/listfiles cannot be used with /p	Review ScanState log or LoadState log for details about command-line errors.
/offline requires a valid path to an XML file describing offline paths	Review ScanState log or LoadState log for details about command-line errors.
/offlinewindir requires a valid path to offline windows folder	Review ScanState log or LoadState log for details about command-line errors.
/offlinewinold requires a valid path to offline windows folder	Review ScanState log or LoadState log for details about command-line errors.
A command was already specified	Verify that the command-line syntax is correct and that there are no duplicate commands.
An option argument is missing	Review ScanState log or LoadState log for details about command-line errors.
An option is specified more than once and is ambiguous	Review ScanState log or LoadState log for details about command-line errors.
By default /auto selects all users and uses the highest log verbosity level. Switches like /all, /ui, /ue, /v are not allowed.	Review ScanState log or LoadState log for details about command-line errors.
Command line arguments are required. Specify /? for options.	Review ScanState log or LoadState log for details about command-line errors.
Command line option is not valid	Review ScanState log or LoadState log for details about command-line errors.
EFS parameter specified is not valid for /efs	Review ScanState log or LoadState log for details about command-line errors.
File argument is invalid for /genconfig	Review ScanState log or LoadState log for details about command-line errors.
File argument is invalid for /genmigxml	Review ScanState log or LoadState log for details about command-line errors.
Invalid space estimate path. Check the parameters and/or file system permissions	Review ScanState log or LoadState log for details about command-line errors.

Error message	Troubleshooting, mitigation, workarounds
List file path argument is invalid for /listfiles	Review ScanState log or LoadState log for details about command-line errors.
Retry argument must be an integer	Review ScanState log or LoadState log for details about command-line errors.
Settings store argument specified is invalid	Review ScanState log or LoadState log for details about command-line errors. Make sure that the store path is accessible and that the proper permission levels are set.
Specified encryption algorithm is not supported	Review ScanState log or LoadState log for details about command-line errors.
The /efs:hardlink requires /hardlink	Review ScanState log or LoadState log for details about command-line errors.
The /targetWindows7 option is only available for Windows XP, Windows Vista, and Windows 7	Review ScanState log or LoadState log for details about command-line errors.
The store parameter is required but not specified	Review ScanState log or LoadState log for details about command-line errors.
The source-to-target domain mapping is invalid for /md	Review ScanState log or LoadState log for details about command-line errors.
The source-to-target user account mapping is invalid for /mu	Review ScanState log or LoadState log for details about command-line errors.
Undefined or incomplete command line option	Review ScanState log or LoadState log for details about command-line errors.
Use /nocompress, or provide an XML file path with /p"pathtoafile" to get a compressed store size estimate	Review ScanState log or LoadState log for details about command-line errors.
User exclusion argument is invalid	Review ScanState log or LoadState log for details about command-line errors.
Verbosity level must be specified as a sum of the desired log options: Verbose (0x01), Record Objects (0x04), Echo to debug port (0x08)	Review ScanState log or LoadState log for details about command-line errors.
Volume shadow copy feature is not supported with a hardlink store	Review ScanState log or LoadState log for details about command-line errors.
Wait delay argument must be an integer	Review ScanState log or LoadState log for details about command-line errors.

12: USMT_ERROR_OPTION_PARAM_TOO_LARGE

- **Category**: Invalid Command Lines

Expand table

Error message	Troubleshooting, mitigation, workarounds
Command line arguments cannot exceed 256 characters	Review ScanState log or LoadState log for details about command-line errors.
Specified settings store path exceeds the maximum allowed length of 256 characters	Review ScanState log or LoadState log for details about command-line errors.

13: USMT_INIT_LOGFILE_FAILED

- **Category**: Invalid Command Lines

Expand table

Error message	Troubleshooting, mitigation, workarounds
Log path argument is invalid for /l	When `/l` is specified in the ScanState command line, USMT validates the path. Verify that the drive and other information, for example file system characters, are correct.

14: USMT_ERROR_USE_LAC

- **Category**: Invalid Command Lines

Expand table

Error message	Troubleshooting, mitigation, workarounds
Unable to create a local account because /lac was not specified	When creating local accounts, the command-line options `/lac` and `/lae` should be used.

26: USMT_INIT_ERROR

- **Category**: Setup and Initialization

Expand table

Error message	Troubleshooting, mitigation, workarounds
Multiple Windows installations found	Listfiles.txt couldn't be created. Verify that the location you specified for the creation of this file is valid.
Software malfunction or unknown exception	Check all loaded .xml files for errors, common error when using `/i` to load the *Config.xml* file.
Unable to find a valid Windows directory to proceed with requested offline operation; Check if offline input file is present and has valid entries	Verify that the offline input file is present and that it has valid entries. USMT couldn't find valid offline operating system. Verify your offline directory mapping.

27: USMT_INVALID_STORE_LOCATION

- **Category**: Setup and Initialization

Expand table

Error message	Troubleshooting, mitigation, workarounds
A store path can't be used because an existing store exists; specify /o to overwrite	Specify `/o` to overwrite an existing intermediate or migration store.
A store path is missing or has incomplete data	Make sure that the store path is accessible and that the proper permission levels are set.
An error occurred during store creation	Make sure that the store path is accessible and that the proper permission levels are set. Specify `/o` to overwrite an existing intermediate or migration store.
An inappropriate device such as a floppy disk was specified for the store	Make sure that the store path is accessible and that the proper permission levels are set.
Invalid store path; check the store parameter and/or file system permissions	Invalid store path; check the store parameter and/or file system permissions.
The file layout and/or file content is not recognized as a valid store	Make sure that the store path is accessible and that the proper permission levels are set. Specify `/o` to overwrite an existing intermediate or migration store.
The store path holds a store incompatible with the current USMT version	Make sure that the store path is accessible and that the proper permission levels are set.

Error message	Troubleshooting, mitigation, workarounds
The store save location is read-only or does not support a requested storage option	Make sure that the store path is accessible and that the proper permission levels are set.

28: USMT_UNABLE_GET_SCRIPTFILES

- **Category**: Setup and Initialization

Expand table

Error message	Troubleshooting, mitigation, workarounds
Script file is invalid for /i	Check all specified migration .xml files for errors. This error is common when using `/i` to load the *Config.xml* file.
Unable to find a script file specified by /i	Verify the location of your script files, and ensure that the command-line options are correct.

29: USMT_FAILED_MIGSTARTUP

- **Category**: Setup and Initialization

Expand table

Error message	Troubleshooting, mitigation, workarounds
A minimum of 250 MB of free space is required for temporary files	Verify that the system meets the minimum temporary disk space requirement of 250 MB. As a workaround, you can set the environment variable `USMT_WORKING_DIR=<path>` to redirect the temporary files working directory.
Another process is preventing migration; only one migration tool can run at a time	Check the ScanState log file for migration .xml file errors.
Failed to start main processing, look in log for system errors or check the installation	Check the ScanState log file for migration .xml file errors.
Migration failed because of an XML error; look in the log for specific details	Check the ScanState log file for migration .xml file errors.
Unable to automatically map the drive letters to match the online drive letter	Check the ScanState log file for migration .xml file errors.

Error message	Troubleshooting, mitigation, workarounds
layout; Use /offline to provide a mapping table	

31: USMT_UNABLE_FINDMIGUNITS

- **Category**: Setup and Initialization

Expand table

Error message	Troubleshooting, mitigation, workarounds
An error occurred during the discover phase; the log should have more specific information	Check the ScanState log file for migration .xml file errors.

32: USMT_FAILED_SETMIGRATIONTYPE

- **Category**: Setup and Initialization

Expand table

Error message	Troubleshooting, mitigation, workarounds
An error occurred processing the migration system	Check the ScanState log file for migration .xml file errors, or use online Help by typing `/?` on the command line.

33: USMT_UNABLE_READKEY

- **Category**: Setup and Initialization

Expand table

Error message	Troubleshooting, mitigation, workarounds
Error accessing the file specified by the /keyfile parameter	Check the ScanState log file for migration .xml file errors, or use online Help by typing `/?` on the command line.
The encryption key must have at least one character	Check the ScanState log file for migration .xml file errors, or use online Help by typing `/?` on the command line.

34: USMT_ERROR_INSUFFICIENT_RIGHTS

- **Category**: Setup and Initialization

Expand table

Error message	Troubleshooting, mitigation, workarounds
Directory removal requires elevated privileges	Sign in as Administrator, and run with elevated privileges.
No rights to create user profiles; log in as Administrator; run with elevated privileges	Sign in as Administrator, and run with elevated privileges.
No rights to read or delete user profiles; log in as Administrator, run with elevated privileges	Sign in as Administrator, and run with elevated privileges.

35: USMT_UNABLE_DELETE_STORE

- **Category**: Setup and Initialization

Expand table

Error message	Troubleshooting, mitigation, workarounds
A reboot is required to remove the store	Reboot to delete any files that couldn't be deleted when the command was executed.
A store path can't be used because it contains data that could not be overwritten	A migration store couldn't be deleted. If you're using a hardlink migration store, you might have a locked file in it. You should manually delete the store, or use `UsmtUtils.exe /rd` command to delete the store.
There was an error removing the store	Review ScanState log or LoadState log for details about command-line errors.

36: USMT_ERROR_UNSUPPORTED_PLATFORM

- **Category**: Setup and Initialization

Expand table

Error message	Troubleshooting, mitigation, workarounds
Compliance check failure; please check the logs for details	Investigate whether there's an active temporary profile on the system.

Error message	Troubleshooting, mitigation, workarounds
Use of /offline is not supported during apply	The `/offline` command wasn't used while running in the Windows Preinstallation Environment (WinPE).
Use /offline to run gather on this platform	The `/offline` command wasn't used while running in WinPE.

37: USMT_ERROR_NO_INVALID_KEY

- **Category**: Setup and Initialization

Expand table

Error message	Troubleshooting, mitigation, workarounds
The store holds encrypted data but the correct encryption key was not provided	Verify that the correct encryption key or keyfile was included with the `/key` or `/keyfile` option.

38: USMT_ERROR_CORRUPTED_NOTENCRYPTED_STORE

- **Category**: Setup and Initialization

Expand table

Error message	Troubleshooting, mitigation, workarounds
An error occurred during store access	Review ScanState log or LoadState log for details about command-line errors. Make sure that the store path is accessible and that the proper permission levels are set.

39: USMT_UNABLE_TO_READ_CONFIG_FILE

- **Category**: Setup and Initialization

Expand table

Error message	Troubleshooting, mitigation, workarounds
Error reading Config.xml	Review ScanState log or LoadState log for details about command-line errors in the *Config.xml* file.
File argument is invalid for /config	Check the command line you used to load the *Config.xml* file. You can use online Help by typing `/?` on the command line.

40: USMT_ERROR_UNABLE_CREATE_PROGRESS_LOG

- **Category**: Setup and Initialization

Expand table

Error message	Troubleshooting, mitigation, workarounds
Error writing to the progress log	The Progress log couldn't be created. Verify that the location is valid and that you have write access.
Progress log argument is invalid for /progress	The Progress log couldn't be created. Verify that the location is valid and that you have write access.

41: USMT_PREFLIGHT_FILE_CREATION_FAILED

- **Category**: Setup and Initialization

Expand table

Error message	Troubleshooting, mitigation, workarounds
Can't overwrite existing file	The Progress log couldn't be created. Verify that the location is valid and that you have write access.
Invalid space estimate path. Check the parameters and/or file system permissions	Review ScanState log or LoadState log for details about command-line errors.

42: USMT_ERROR_CORRUPTED_STORE

- **Category**:

Expand table

Error message	The store contains one or more corrupted files
The store holds encrypted data but the correct encryption key was not provided	Review UsmtUtils log for details about the corrupted files. For information on how to extract the files that aren't corrupted, see Extract files from a compressed USMT migration store.

61: USMT_MIGRATION_STOPPED_NONFATAL

- **Category**: Non-fatal Errors

Expand table

Error message	The store contains one or more corrupted files
Processing stopped due to an I/O error	USMT exited but can continue with the `/c` command-line option, with the optional configurable **<ErrorControl>** section or by using the `/vsc` command-line option.

71: USMT_INIT_OPERATING_ENVIRONMENT_FAILED

- **Category**: Fatal Errors

Expand table

Error message	Troubleshooting, mitigation, workarounds
A Windows Win32 API error occurred	Data transfer has begun, and there was an error during the creation of migration store or during the apply phase. Review the ScanState log or LoadState log for details.
An error occurred when attempting to initialize the diagnostic mechanisms such as the log	Data transfer has begun, and there was an error during the creation of migration store or during the apply phase. Review the ScanState log or LoadState log for details.
Failed to record diagnostic information	Data transfer has begun, and there was an error during the creation of migration store or during the apply phase. Review the ScanState log or LoadState log for details.
Unable to start. Make sure you are running USMT with elevated privileges	Exit USMT and sign in again with elevated privileges.

72: USMT_UNABLE_DOMIGRATION

- **Category**: Fatal Errors

Expand table

Error message	Troubleshooting, mitigation, workarounds
An error occurred closing the store	Data transfer has begun, and there was an error during migration-store creation or during the apply phase. Review the ScanState log or LoadState log for details.
An error occurred in the apply process	Data transfer has begun, and there was an error during migration-store creation or during the apply phase. Review the ScanState log or LoadState

Error message	Troubleshooting, mitigation, workarounds
	log for details.
An error occurred in the gather process	Data transfer has begun, and there was an error during migration-store creation or during the apply phase. Review the ScanState log or LoadState log for details.
Out of disk space while writing the store	Data transfer has begun, and there was an error during migration-store creation or during the apply phase. Review the ScanState log or LoadState log for details.
Out of temporary disk space on the local system	Data transfer has begun, and there was an error during migration-store creation or during the apply phase. Review the ScanState log or LoadState log for details.

Data collection

If you need assistance from Microsoft support, we recommend you collect the information by following the steps mentioned in Gather information by using TSS for deployment-related issues.

Related articles

User State Migration Tool (USMT) troubleshooting

USMT log files

USMT resources

Article • 01/29/2025 • Applies to: ✅ Windows 11, ✅ Windows 10

USMT online resources

- ADK Release Notes.
- Microsoft Visual Studio
  - The User State Migration Tool (USMT) XML schema (the `MigXML.xsd` file) can be used to validate the migration **.xml** files using an XML authoring tool such as Microsoft Visual Studio.

    For more information about how to use the schema with an XML authoring environment, see the environment's documentation.
- Ask the Directory Services Team blog.
- Forums:
  - Microsoft Deployment Toolkit forum.
  - Configuration Manager Operating System Deployment forum.

Related articles

User State Migration Tool (USMT) overview articles.

Feedback

Was this page helpful? Yes No

Provide product feedback