Microsoft

Optimizing Windows 10 update adoption

Microsoft

Contents

Microsoft

Overview

When we talk to IT administrators at conferences or direct engagements, we are often asked about maximizing velocity when deploying Windows 10 monthly security updates in the enterprise and how to deploy feature updates as efficiently as possible. The feedback has been consistent: the tradeoffs of various configuration settings, device health and system resource availability are not readily apparent, and the impacts of these choices are not entirely clear.

IT administrators tell us that they want to get their devices protected as quickly as possible – especially in a heightened security landscape – but want to minimize disruption to their organization, and they struggle with the right mix of settings and how to monitor their success.

Some common goals are to:

- Reduce the cost of approving, deploying, and monitoring updates.
- Manage application compatibility within the organization's ecosystem.
- Find the right tradeoffs to protect devices, while minimizing disruption to the workforce.
- Manage the infrastructure configurations necessary to support rapid update velocity, including finding the right way to address devices that are rarely connected to the enterprise.

Protected and productive

There has always been a tension between the need for timely software update compliance and the desire to keep workers productive. While the Chief Security Officer may wish to see a fully updated network within seven days of a software update, the reality is that deployment of said update has an associated cost for users and very few companies can afford to push an update on an entire workforce in the middle of a single working day unless it is a dire emergency.

Given the competing goals of a protected and a productive work force, you may find that you may need to make choices that are less than the best possible selection for maximizing update velocity in favor of an experience more aligned with your business's productivity needs. Microsoft makes these same choices in the default Windows behavior to best optimize the end user experience while meeting the compliance goals that you, as the administrator, specify.

How to use this document

To help you better understand the policy settings that impact velocity, how to monitor your deployments in order to continue to improve processes, and find information on deployment strategies, we present the following topics:

- **Tuning update policies in Windows Update for Business and WSUS for increased velocity.** If you have devices configured to receive updates from Windows Update for Business or Windows Server Update Services (WSUS), explore the tradeoffs between velocity and productivity—and better understand the impact of policy and device settings to the devices across your organization. In addition, find out how to create a successful update process for low-activity devices.

- **Tuning devices for increased velocity.** If you use any update management technology, including Microsoft Endpoint Configuration Manager, learn more about the system policies and configurations that impact update success. This guide will walk you through infrastructure optimization, adjusting your network utilization choices, and addressing network congestion.

- **Monitoring and enforcement.** Fine-tune your update deployment processes by diving into data to discover what's working and what still needs to be addressed. Learn techniques for troubleshooting and find ways to continually improve the update process in your organization.

- **Deployment strategies.** If you are new to Windows deployment practices, this is a great place to begin. Learn about adopting a service management mindset, find recommended practices for feature update deployments, and discover tools that can assist you in identifying the right diversity of devices to help make feature update (and security update) deployments more efficient.

- **Policy setting reference.** Get a handy checklist you can reference when applying the policies and settings recommended in this document.

By following the best practices outlined in this document, we expect your update velocity to increase while simultaneously keeping your workforce productive and satisfied.

Microsoft

Tuning update policies in Windows Update for Business and WSUS for increased velocity

If you have devices that use Windows Update for Business or Windows Server Update Services (WSUS) to manage updates, there are several policies that are of interest. In order to maximize update velocity while remaining mindful of user productivity impact, Microsoft suggests a specific suite of administrative policies with recommended values, as well as a set of policies we recommend you do *not* set. We have ordered these policies with those where our data has shown the highest impact on velocity first.

How Windows updates work

There are four phases to the Windows Update process:

- **Scan.** A devices checks the Microsoft Update server or your 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 guidelines (e.g. Group 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.

- I**nstall.** 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 phase a device runs the previous version of the software.

At each stage of the process, there are opportunities to increase velocity via policies and settings and our recommendations follow.

Compliance deadlines

Setting Compliance Deadlines is the most important policy that every enterprise who cares about achieving reliable update velocity should set. Deadline policies are the supported mechanism for administrators to communicate their intent around how quickly the update components of Windows should reliably complete. These Windows components adapt their behavioral heuristics based on these deadlines in order to attempt to meet the stated deadline.

Also, when specifying deadlines, the Windows components can make tradeoffs between user experience and velocity in order to meet your stated deadline. For example, it can prioritize user experience well before the deadline approaches, and then prioritize velocity as the deadline nears, while still affording the user some control.

By setting compliance objectives around how quickly you want your organization's devices to fully apply quality and feature updates, you will be able to measure progress – release over release – and tweak policies over time to meet your evolving business requirements, ultimately increasing velocity.

Setting deadlines

Beginning with Windows 10, version 1903 and with the August 2019 security update for Windows 10, version 1709 and above, a new policy was introduced: Specify deadlines for automatic updates and restarts, replacing previous deadline-like policies.

Revisions prior to August 2019 started enforcing deadlines once the device reached a "restart pending" state for an update; whereas this 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).

> **Important:** If you use the new Specify deadlines for automatic updates and restarts setting in Windows 10, version 1903, you must *disable* previous Auto Restart Deadline policies as they may conflict.

We recommend you set up deadlines as follows:

- Quality update deadline, in days: 3
- Feature update deadline, in days: 7

Notifications are automatically presented to the user at appropriate times, and users may 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, as they are automatically configured with appropriate defaults, unless you have kiosks or digital signage, in which case you can find documentation here.

While three days for quality updates and seven days for feature updates is our recommendation, you may 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 will be unable to determine when Microsoft published the update, so it will not be able to enforce the deadline. Learn more about low activity devices.

Setting grace periods

If a device has not completed the update by the deadline, the grace period specifies the number of days the device has for Windows to find a minimally interruptive automatic restart time before the restart is enforced. This is especially useful in cases where a user has been away for many days (e.g. on vacation) so that the device will not be forced into an immediate update upon the user's return. Again, this is configurable based on what a reasonable grace period might be for your organization.

We recommend you set the following:

- Grace Period, in days: 2

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

Allowing Windows to choose the least disruptive time for reboot

Windows has heuristics based on user interactions that dynamically identifies the least disruptive time for automatic restart. To take advantage of this feature, ensure **ConfigureDeadlineNoAutoReboot** is set to **Disabled**.

Accounting for low activity devices

What if the compliance deadline is set, the grace period has passed, and devices are still not updating? We've found that many devices are not being used enough or are not connected to the internet for long enough to obtain and install an update.

Windows typically requires a device to be active and connected to the internet for at least 6 hours, with at least 2 of those hours being continuous, in order to successfully complete a system update. The device may 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 machine before active hours ends and the device cannot comply with the deadline.

The following section discusses Group Policy and mobile device management (MDM) settings that – in conjunction with Compliance Deadlines – ensure devices are actually available to take updates during the compliance period.

Set active hours

Active hours identify the period of time when a device is expected to be in use, and restarts will occur outside of these hours. Windows 10, version 1903 introduced a feature called intelligent active hours, whereby the system learns active hours based on a user's activities, allowing the device to set active hours based on observed user behavior and find the best time to update the device, rather than having you as an administrator make one-size-fits-all decisions for your organization, or allow the user to select active hours that minimize the windows of time where the OS can update.

Important: If you used **Configure Active Hours in Group Policy** in previous versions of Windows 10, these options must be set to **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 may delay an update indefinitely if a user is always either logged in or shut down. Instead, we recommend setting the following polices to **Disabled**:
    - Turn off auto-restart during active hours
    - No auto-restart with logged on users for scheduled automatic updates
- **Limit restart delays**. Again, by using compliance deadlines, your users will receive notifications that updates will occur, and we recommend that this policy be set to **Disabled**, allowing 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**. By having users approve or engage with the update process outside of the deadline policies decreases velocity and increases risk. These policies should be set to **Disabled**:
    - Update/RequireUpdateApproval
    - Update/EngagedRestartDeadline
    - Update/EngagedRestartDeadlineForFeatureUpdates
    - Update/EngagedRestartSnoozeSchedule
    - Update/EngagedRestartSnoozeScheduleForFeatureUpdates
    - Update/EngagedRestartTransitionSchedule
- **Configure automatic update**. By properly setting policies to configure automatic updates, velocity is increased by having clients contact the WSUS server so it can manage them. We recommend that this policy be left as **Not Configured**. However, if there are reasons to provide values, ensure that you are setting downloads to install automatically by setting the Group Policy to **4** or if you're using Microsoft Intune, setting the value to Reset to **Default**. In Group Policy, to create the best opportunity for an update, we recommend setting:
    - "Scheduled install day" to "0 – Every day"
    - "Scheduled install time" to "Automatic"
    - Select to update "Every week"

- **Allow auto Windows Update to download over metered networks**. Given that a significant number of devices primarily use cellular data and do not have wi-fi access, consider allowing users to automatically download updates from a metered network. While the default setting does not allow download over a metered network, setting this value to **1** can increase velocity by enabling users to get their updates whether they are connected to the internet or not, provided they have cellular service.

Important: Older versions of Windows do not support intelligent active hours. If your device is running 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 may specify a maximum active hour range and this range will be counted from the active hours start time. We recommend setting this value to **10**.
- **Schedule update installation**. Within **Configure Automatic Updates**, there are two ways to control a forced restart after a specified installation time. (When setting the schedule update installation, do not enable both settings as 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**. 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 install time for a restart. We do not recommend you set this to **Disabled** as it may conflict with active hours.

Power policies

Now that compliance and reboot policies are set, it's time to see if devices are actually available during non-active hours to take an update. Many organizations have set "green" power policies which have interfered with devices taking updates during non-active hours. We strive to set a balance between security and eco-friendly configurations when considering power policies for devices in our organization and we recommend the following settings to achieve what we feel are the appropriate tradeoffs.

Setting power policies through CSP

To the user, a device is either on or off, but for Windows, there are states that will allow an update to occur (active) and states that do not (inactive); and 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 looks at before deciding to start an update.

While power settings have default states on Windows devices, you may override these, preventing users from changing them, in order to ensure that devices are available for updates during non-active hours.

Note: Educating your users on keeping devices plugged in during non-active hours is a way to ensure that devices can be updated during compliance timeframes. Even with the best policies in place, a device that is not plugged in will not be updated, even in sleep mode. And with smart power policies, you can be sure that you are balancing green initiatives with keeping the devices on your network safe.

Here are recommendations for power management settings:

- **Sleep mode (S1 or S0 Low Power Idle/Modern Standby).** When a device is in sleep mode, the system appears to be off but if an update is available, it can wake the device up 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 is not 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 does not 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 is 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 very low and the system cannot wake up without user intervention, like pressing the power button. If a device is in this state, it cannot 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 will be delayed until the update is completed.

  Note: This does not apply to devices supporting 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'.

  The default timeout on devices supporting traditional sleep is set to 3 hours, and we recommend that you do not reduce these policies in order to allow Windows Update the opportunity to reboot the machine before sending the device into hibernation:

  - Power/HibernateTimeoutOnBattery

- Power/HibernateTimeoutPluggedIn

Disabling conflicting or legacy policies

With each release of Windows 10, there are new policies that are introduced that ultimately 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 above or we backport the policy to make it available on earlier versions. For example, Compliance Deadline was released with Windows 10, version 1903 and is also available in Windows 10, version 1803 and Windows 10, version 1809 beginning with the August 2019 monthly quality update.

Important: If you are using Group Policy, note that we do not update the old ADMX templates and you must utilize the newer (1903) ADMX template in order to utilize the newer policy. Additionally, if you are using an MDM tool (Microsoft or third-party), you are not able to utilize the new policy until it is made available in the tool interface.

As administrators, you have set up and expect certain behaviors, so we expressly do not remove older policies as they were set up for your particular use cases. However, if you set a new policy without disabling a similar older policy, you may have conflicting behavior and your update process may 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 via Group Policy with a differing value.

If you find that velocity is not as high as you expect or if some devices are slower than others, it may be time to clear *all* polices and settings and specify only the recommended update policies outlined in this document. See the Policy and settings reference for a consolidated list of recommended polices to increase update velocity.

The following are a list of policies that you may want to disable as they may decrease velocity or there may be better policies to use that may conflict:

- **Defer Feature Updates Period in Days**. For maximum velocity, we recommend that this be set to **0** (no deferral) so that the feature update can complete and monthly security updates will be offered again. Even if there is an urgent quality update that must be quickly deployed, it is best to use **Pause Feature Updates** (see below) rather than setting a deferral policy. You may choose a longer period if you do not wish to stay up to date with the latest feature update.

- **Defer Quality Updates Period in Days**. To minimize risk and maximize velocity, the maximum time you might want to consider while evaluating the update with a different ring of devices is 2-3 days.

- **Pause Feature Updates Start Time**. Set to **Disabled** unless there is a known issue where time is needed for a resolution.

- **Pause Quality Updates Start Time**. Set to **Disabled** unless there is a known issue where time is needed for a resolution.

- **Deadline No Auto Reboot**. Default is **Disabled** – **Set to 0** – and we recommend that devices automatically try to restart when an update is received. As discussed above, Windows has heuristics based on user interactions that dynamically identifies the least disruptive time for reboot. (See our recommendation in active hours.)

There are additional policies are no longer supported or have been superseded and may be decreasing velocity. Go to Policies and settings reference guide – Policies to disable or not configure for more information.

Distribution point hygiene

Once these policies are in place, you will be able to monitor to see if there are categories of devices that are active but unable to update (see Monitoring and enforcement) and whether issues are encountered in the stack. Here are some common errors we see around devices unable to receive updates due to the stack:

- **Servers are unavailable**. It could be that a group of devices are pointing to a server that is offline or even to a WSUS server that is no longer in existence. Gather the list of servers across your ecosystem that you have configured devices to connect to and ping them to be sure they are live.

- **Update not approved in WSUS**. Have you approved the update on every WSUS server in your organization? You'd be surprised at how often this step is missed and devices are not taking the update because it is not approved for deployment. When Dual Scan is intentionally disabled via the "DisableDualScan" policy the default update service for Windows Update client switches back to WSUS

and all updates belonging to Windows and non-Windows product family are evaluated and offered from WSUS during scan. Ensure that you have approved the necessary updates. Note that this configuration does not preclude the offering of updates from the online Windows Update service. Any manual scan against the online Windows Update service will offer updates from those services with appropriate deferrals and pause applied to those updates.

Blocked devices

Once you've ensured that policies are in place and devices are actually able to reach an update, there may be problems with the devices themselves that prevent them from updating. While the following information is for Windows Update for Business and WSUS systems, additional information can be found in Unhealthy devices.

Here are some common issues that may be keeping devices from being updated:

- **Feature updates obstructing quality updates.** If a device is offered a feature update, it takes this update in favor of any monthly update, to get the latest and greatest features as quickly as possible. It is very important to monitor the progress of a feature update deployment and ensure it does not overlap with a security update time (second Tuesday of the month). If the feature update runs into trouble installing, it can prevent the installation of an important security update. If you have such devices, work to resolve them as quickly as possible and consider deploying a feature update pause or feature update deferral policy (For the latter, we recommend 365 days.) to give the device time to take the monthly security update. Then you can un-pause or remove the deferral at your leisure.

- **Recurring installation error**. If a device is throwing a recurring installation error, it may have bad data staged in its Windows software distribution folder. See "Clean up the software distribution folder" under Unhealthy devices.

Tuning devices for increased velocity

Whether you manage updates with Configuration Manager or WSUS or Windows Update for Business, there are several things you can do to tune individual systems and network configuration to maximize update velocity while remaining mindful of user productivity impact.

Ensuring updates are available

- **Enable update services on the clients**. Ensure every device is running all the services on which Windows Update relies. Sometimes users or even malware can disable the services Windows Update requires in order to function correctly. Make sure the following services are running:
    - Background Intelligent Transfer Service
    - Background Tasks Infrastructure Service
    - BranchCache (if you rely on this feature for update deployment)
    - ConfigMgr Task Sequence Agent (if you rely on Config Manager for deployment)
    - 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 Center Service
    - Windows Time Service
    - Windows Update
    - Windows Update Medic Service
- **Network configuration.** Ensure that devices can get to necessary Windows Update endpoints in the firewall.

- **Optimize download bandwidth**. Use Delivery Optimization to leverage peer network sharing or Microsoft Connected Cache. Learn more in Bandwidth optimization/peer-to-peer deployment.

Unhealthy devices

Some devices face challenges that create problems in successfully apply updates – we call these unhealthy devices – and you can take actions to make these devices more successful in taking updates.

- **Low disk space**. Quality updates require a minimum of 2GB to successfully install; feature updates require between 8 and 15 GB depending upon the configuration. Proactively, on Windows 10, version 1903 and later you can use the reserved storage feature in Windows 10 (for wipe and loads, rebuilds, and new builds) to avoid running out of disk space. If you find a group of already-deployed devices that do not have enough disk space, cleaning up log files – and asking users to clean up data if necessary – can often bring these devices into compliance. (See Monitoring and enforcement for more on detecting these issues.) Deleting the following files are a good place to start:

    - C:\Windows\temp
    - C:\Windows\cbstemp (this file may be necessary for update failure investigations)
    - C:\Windows\WindowsUpdate.log (this file may be necessary for update failure investigations)
    - C:\Windows.Old (these files should automatically clean up after 10 days or may request 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 the client, with administrator privileges, or use Group Policy settings.

    - Clean up the Windows Store Cache:
        - C:\Windows\sytem32\wsreset.exe
    - Optimize the WinSxS folder on the client machine using DISM:
        - Dism.exe /online /Cleanup-Image /StartComponentCleanup
    - Compact the OS:
        - Compact.exe /CompactOS:always
        - Remove Windows Features on Demand the user does not need. See Features on Demand for more guidance.
    - Move Windows Known Folders to OneDrive. See Use Group Policy to control OneDrive sync settings for more details.

    - Clean up the Software Distribution folder. Try deploying the following script to these devices that runs in a batch file on each affected machine to reset the download state of Windows Updates:

        - 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 may prevent devices from updating successfully. Utilize Desktop Analytics to identify drivers and applications that need attention. You can also check for known issues in order to take any appropriate action. Deploy any updates from the vendor(s) for any problematic application or driver versions to resolve issues. See Monitoring and enforcement for more information on finding issues.

- **Corruption**. In rare circumstances, a device encountering repeated installation errors may be corrupted in a way that prevents the system from applying a new update on top of the old one, and the Component Based Store may have to be repaired from another source. Learn how to use the System File Checker tool to repair missing or corrupted system files.

Bandwidth optimization/peer-to-peer sharing

Another challenge in maximizing update velocity may be alleviated by looking at minimizing Internet bandwidth utilization on your network. You may find that updates coming from the cloud consumes bandwidth and may incur additional download costs as the same update is downloaded to each individual client. Contention for bandwidth, either at the WAN access point or at the individual PC, may introduce user-perceptible performance degradation, or may throttle down the velocity of an update reaching an intended device.

Microsoft has several network-optimization options, depending on your business, and has introduced new tools to best deliver updates to devices across your network. We highly recommend Delivery Optimization above other option, as we believe that by implementing Delivery Optimization peer-to-peer sharing, and leveraging its rich bandwidth throttling configuration, you will most likely find increased velocity, reduction of bandwidth, and ultimately decrease in cost.

What is Delivery Optimization?

Delivery Optimization is a reliable downloader and a self-organizing distributed cache that allows clients to receive file packages from alternate sources (such as other peers on the network) in addition to the traditional internet-based servers. Windows updates can contain large packages and downloading and distributing these files can consume quite a bit of network resources on the devices receiving them. Delivery Optimization reduces bandwidth consumption by sharing the work of downloading file packages among multiple devices in your deployment.

With Delivery Optimization, clients can be configured into groups, allowing organizations to identify devices that are possibly the best candidates to fulfill peer-to-peer requests. Delivery Optimization can significantly reduce the amount of network traffic to external Windows Update sources as well as the time it takes for clients to retrieve the updates, thus increasing velocity while minimizing friction.

But Delivery Optimization is not just peer-to-peer; Delivery Optimization performs *all* the downloads that are executed whether or not you are using peer-to-peer. In other words, there are additional options we expose that are purely about managing bandwidth that may be useful, even without peer-to-peer options. For example, you can throttle bandwidth at a particular time of day, or throttle just background traffic if needed.

Furthermore, Delivery Optimization offers a feature called **Microsoft Connected Cache**, which allows you to configure a dedicated in-network transparent cache for files requested through Delivery Optimization, working in parallel with peers. By setting up the Connected Cache at the WAN or Internet access point, you can minimize file downloads across your WAN or across a limited-bandwidth Internet connection. Learn more about Microsoft Connected Cache.

Best practices for Delivery Optimization

Delivery Optimization has many settings to fine-tune behavior (See Delivery Optimization reference for a comprehensive list), but to get started, we have identified some key use-cases and policies with corresponding recommended Delivery Optimization settings.

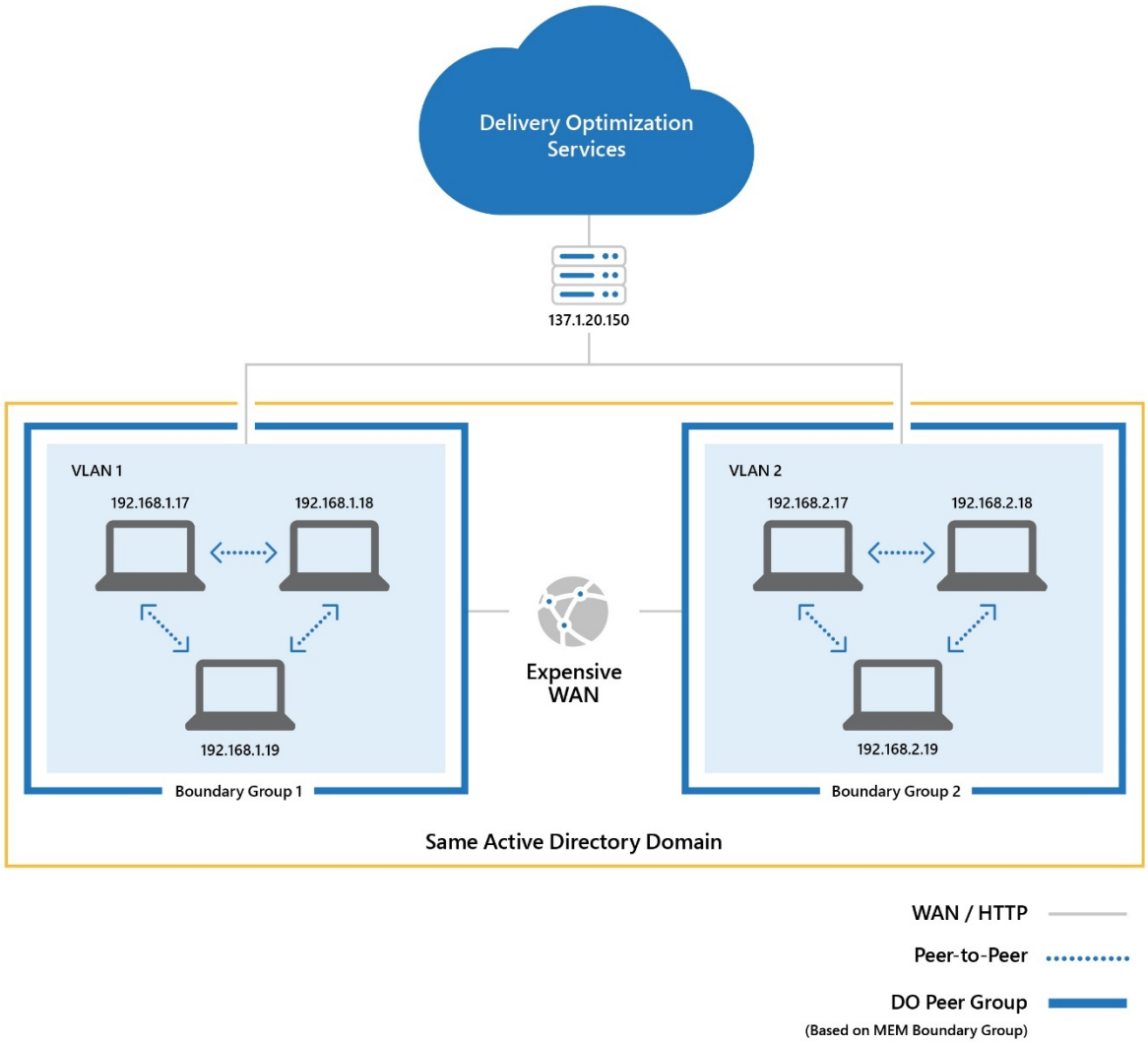
If your network is a hub-and-spoke topology and devices are managed by Configuration Manager, look at the Download Mode policy.

The default setting for the **Download Mode** policy is **1**, which means that clients will attempt to connect using their private IPs, to peer devices on the same LAN (same public IP). In other words, when **Download Mode** is set to **1**, all devices with the same public IP can connect using their private IP.

To illustrate, let's say we have a group of devices in our Northwest site, but in fact, some devices are in Seattle and some are in Portland. You would not want devices connecting to each other across locations.

One of the most useful options to address this is to group peers (devices) based on a boundary group that is already defined in Microsoft Endpoint Manager. So, with Microsoft Endpoint Manager, by enabling the boundary group option, you can now apply a policy whereby Delivery Optimization looks at (Group ID) which assigns all machines in that boundary group to a single unified group based on the device location.

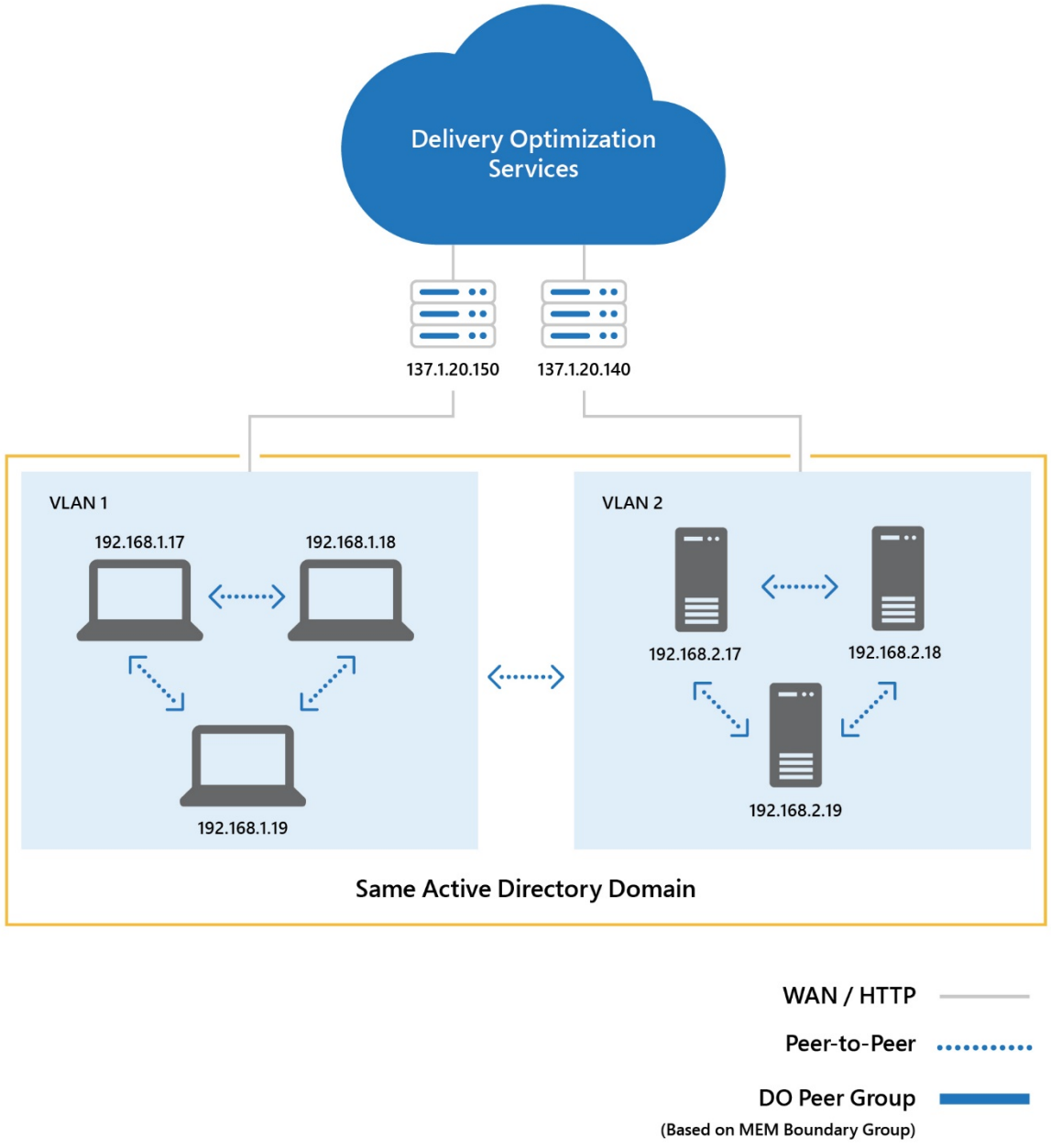
And if you have devices that move between locations, Configuration Manager Agent reapplies the group association based on the boundary group accordingly allowing the device to take the update based on the optimal location. In other words, Configuration Manager Agent "stamps" clients with the appropriate Delivery Optimization group ID when they roam.



Learn more about hub-and-spoke topology with boundary groups.

Once you enable the Boundary Group feature in Configuration Manager, it also sets the **Download Mode** policy setting to **2** to allow downloads to occur from devices within the boundary group.

If you do not use Configuration Manager, you can set Group Download Mode to 2 and then create peer groups dynamically using AD Sites, DHCP, DNS Domain Suffix and Custom (via script that sets the GroupID policy), as illustrated in the following image.



For further instructions, read the Delivery Optimization reference.

If your network is a hub-and-spoke topology, consider deploying Microsoft Connected Cache.

If you configure Microsoft Connected Cache at the network access point, you can avoid duplicative update requests all the way back to your hub. If you are a Configuration Manager customer, you can deploy Microsoft Connected Cache and leverage your distribution points to transparently cache content that would otherwise be downloaded from the cloud. You can still leverage Delivery Optimization peer-to-peer in parallel to Microsoft Connected Cache and boundary groups by enabling that feature.

Read the Delivery Optimization reference for further instructions.

If your sites have more than thirty devices, change the Minimum Peer Caching Content File Size policy.

By default, the peer-to-peer download limit is set at 50MB. But these days, since even laptops come with large internal hard-drive capacity, very little space is being used on each device so there may be available room to enable Delivery Optimization for downloads over 10MB, especially if you have a large number of devices in your network.

> **Note:** Delivery Optimization automatically checks and purges cached content that is not needed. As downloads happen, Delivery Optimization continuously prunes what is written in the cache, so in general, caching content on devices with low disk space is not an issue.

Learn how to set the Minimum Peer Caching Content File Size.

If your organization has a large number of mobile devices, adjust settings in the Allow Uploads on Battery Power policy.

Oftentimes (and by default), peer-to-peer downloads are disabled for machines on battery power but with higher-end devices with the latest wireless network technology and higher capacity, this may not be necessary anymore. By configuring Delivery Optimization to upload content when the battery percentage is above a certain percentage (At Microsoft, this is set to above 60%.), you can allow uploads on battery power while also limiting use to prevent drain.

Learn more about allowing uploads on battery power.

If your labs include AC-powered devices, update the Content Expiration policy.

In a lab scenario, there are typically a large number of devices that have constant power and free disk space. Take advantage of these devices as an upload source by allowing content to be cached over a longer period.

You can configure these devices to store updates for a maximum of 7 days instead of the default which is up to 72 hours; and organizations doing ring deployments usually set the cache expiration for 7-14 days to match the longest ring deferral.

Learn more about setting the maximum cache age.

If you are comfortable with a short delay, consider extending the wait time for peer-to-peer.

The idea behind a hybrid peer-to-peer sharing model is that when a peer is not available, the device goes back to the cloud via http to ensure we always complete the download. When we distribute a security update, we want to be sure the content gets to where it needs to go. That said, if you are comfortable with a short delay, you can set a policy that says, "Wait! Before you go to the cloud to pull content, wait for a peer to become available because if that peer shows up, you'll most likely get better performance overall. And if a peer does not show up in the allotted time, use HTTP. We recommend delaying 5 minutes for using HTTP in the background and 1 minute in the foreground. Learn more about delaying the use of HTTP in the background or foreground.

Addressing network congestion

In talking with customers, especially those in the education sector, we discovered that many organizations do not have a big network pipe or that Wi-Fi routers are congested and do not keep up with peer-to-peer traffic. Administrators are also concerned about shared bandwidth with mission-critical devices where performance is crucial.

To address these scenarios, Delivery Optimization automatically uses "Low Extra Delay Background Transfer" (LEDBAT)—a latency optimized, network congestion control provider—and enables it for peer-to-peer connections with LEDBAT-enabled devices on the local network. LEDBAT identifies network congestion in milliseconds and increases the client's intelligence to use the network resources wisely by backing off peer-to-peer uploads and avoiding impact on higher priority network traffic. LEDBAT allows the peer-to-peer traffic to ramp up quickly to consume network bandwidth when the network is underutilized.

While Delivery Optimization is automatically LEDBAT enabled, if you are using WSUS as your local content cache or Configuration Manager as a Distribution Point, LEDBAT is also available for the hosting server and we recommend you use it in order to take full advantage of the benefits of this capability.

Learn more about LEDBAT and Configure Distribution Points.

Delivery Optimization resources

- Recommended settings for Delivery Optimization
- Delivery Optimization for Windows 10 updates
- Set up Delivery Optimization for Windows 10 updates
- Delivery Optimization reference
- Policy CSP – DeliveryOptimization

Other optimization options

While all of the following peer-to-peer solutions can be used in parallel, we highly recommend Delivery Optimization above other options. That said, we outline your options here.

- **Recommended: Delivery Optimization** is the reliable built in download and peer-to-peer distribution technology in Windows 10 whereby clients can source content from other devices on their local network that have already downloaded the updates.
- **Peer Cache for Configuration Manager clients** is another peer-to-peer option that can assist in managing deployment of content to remote locations. Learn more about Peer cache.
- **BranchCache** is an optimization technology that is included with some editions of Windows Server and Windows 10, and while BranchCache has been widely adopted over the years, we highly recommend moving to Delivery Optimization as Windows 7 has reached end of service. If you would like more information on Branch Cache, you may find it here.

	Delivery Optimization	Peer cache	BranchCache
Supported content type	Windows and security updates, Drivers, Windows Store for Business apps, Microsoft Intune Win 32, Office ProPlus installation and updates	All Configuration Manager content types including images (no policies)	All Configuration Manager content types including images (no policies)
Supported across subnets	Yes	Yes	No
Discovery of a peer source	Automatic	Manual (client agent setting)	Automatic
Peer discovery	Via Delivery Optimization cloud service (requires Internet access)	Via management point (based on client boundary groups)	Multicast

Other optimization resources

- Optimize Windows 10 update delivery
- Delivery Optimization for Windows 10 updates
- Configure BranchCache for Windows 10 updates

Monitoring and enforcement

Now that you have revised policies and processes for Windows 10 updates, it's time to quantify your work and use that data to inform your policies and processes moving forward. By strategically monitoring the update process, you will most likely learn of any issues that may arise, giving you data to actively remediate any problems that may be blocking users from successfully installing their updates, and longer-term, monitoring the effect on changes to policies and settings that can help you find which fixes are having an impact, and which are not.

There are two main types of data that you can use to monitor update velocity within your enterprise:

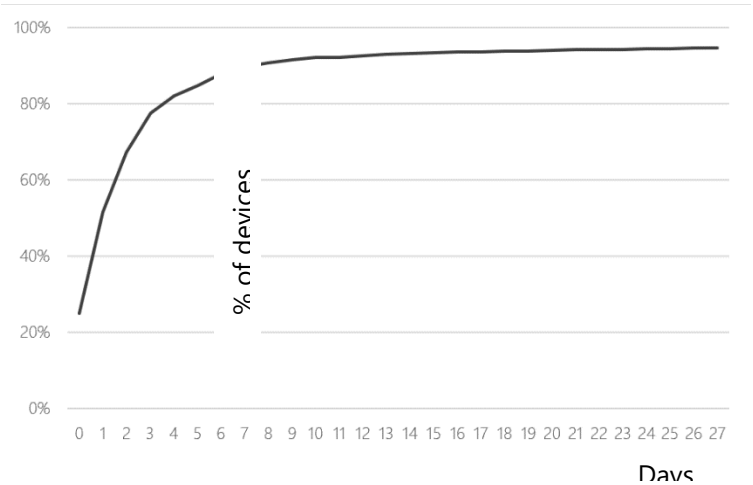
- Number of devices that have successfully updated compared to your velocity goals
- Break down of issues or behaviors impeding velocity across your ecosystem

The following are best practices around monitoring updates and using data to take actionable steps to continue to increase update velocity, while being mindful of user productivity.

Monitoring strategies

At the very highest level, one of the most important metrics for velocity is the percentage of devices that were successfully able to install a particular update within X days of when it was offered. At Microsoft, we measure X at 7 days, 14 days and 28 days after the update was offered for routine monthly updates and we measure daily for urgent updates, which generally happen two or three times per year.

In the healthy consumer Windows 10 population, we see an adoption curve similar to the one shown below. Note that this curve does not include devices with deadline policies set, but is shaped primarily by the activity pattern of the devices:



As you can see, we expect an immediate trend upward in the first 7 days, and after 14 days, we typically see an update rate of 93% among updatable devices. After 28 days, we generally see an adoption rate of 95%.

When benchmarking update velocity for your organization, if you have set a quality update deferral, add your deferral days to these benchmarks to set your minimum expectations of what your velocity should look like.

We believe that if you follow the recommendations in this document, you should exceed this rate of adoption for your organization, and if you do not, you should continue looking for additional optimizations.

At Microsoft, we are always looking at the remaining "not updated" devices to catch devices that become unhealthy over time and as we continue to optimize in order to increase velocity, we investigate to see why they are not taking updates as expected. We then take these devices and categorize them using our analysis taxonomy and consider if any remediation actions are required.

It is important to recognize that a device may have multiple issues that can prevent it from updating, and oftentimes issues must be addressed in a particular sequence. We recommend looking at device issues in the following priority order:

- **Low activity –** Devices that do not have at least 6 hours of active use while connected to the internet, with at least 2 of those hours being continuous. These devices can be addressed by increasing scan frequency and by adjusting power policies.
- **Unsuccessful scan –** Devices that have not successfully completed a Windows Update scan to completion. These devices may either be encountering an error, or timing out on the scan. If you cannot resolve the error, you may need to re-image these devices. Learn more about addressing error codes.
- **Compatibility –** Devices that have a known compatibility issue with hardware, apps, or drivers. These devices would encounter errors if they install the upgrade. Use Desktop Analytics to ensure the devices on your network are compatible. Learn more about compatibility safeguards by checking for known issues.
- **Not offered –** Devices where the source of the update has not approved the update package for distribution. Typically, these would be bad update endpoints or a failure to approve the update in a WSUS or Configuration Manager environment. Learn more about ensuring updates are available.
- **Policy –** Devices where a policy that you have set on the client is interfering with the update or where the default update stack behavior is deferring the update but that behavior is controllable via a policy that you have not set. Learn more about recommended policies.
- **Disk space –** Devices that have insufficient disk space to take an update (2GB+ for monthly security updates, 10GB+ for feature updates).
- **Problems –** Devices that tried to install but encountered an error either in the update stack or in the Windows setup program. Learn more about addressing update errors.
- **In progress –** Devices that are making progress installing but have not yet reached the reboot state or encountered an error and moved back to the Problems category.

- **Waiting for user restart –** Devices that have scanned, downloaded, and installed but are still waiting to commit and restart in order to apply the update. You can force these devices to restart in order to quickly reach your compliance goals.

Categories listed first may hide the presence of an issue lower on the list; for example, a policy change generally will not affect the ability for a device to update due to insufficient activity.

Once devices are categorized, we look at the sub-group of devices in order to find possible issues, and then identify work to remedy those devices' conditions.

By systematically monitoring your devices, you'll be able to continue making small changes that can remedy issues impacting velocity in meaningful ways.

Monitoring your deployment with Desktop Analytics

As you are preparing to deploy, during pilot, and after you have deployed to production, Desktop Analytics may provide some quick, actionable feedback to assist you in troubleshooting issues that may arise.

> **Note:** Desktop Analytics is intended for use primarily with Configuration Manager.

By looking at the Compatibility Assessment, you will find low, medium, and high risks associated with your deployment and see if any assets have been fully or partially removed due to possible compatibility issues. In addition, you'll find additional insights around possible problems with apps and drivers that may be blocking devices from taking an update.

Health Status Monitoring looks at factors like percentage of devices with crashes, usage characteristics, and more, providing a topline look at how your upgrade is going.

While Desktop Analytics can give you an overview of the issues that have to do with compatibility, you can dive deeper into your organization's deployment using Update Compliance.

Using Update Compliance

We recommend using Update Compliance to provide you with detailed deployment data for Windows 10 security, quality and feature updates. In addition, you'll be able to access data to highlight how you can show bandwidth utilization savings for devices that are configured to use Delivery Optimization. You may also access the data that underlies Update Compliance directly in Log Analytics with additional features like custom searches, data visualizations, and alerts.

Update Compliance provides some key features for monitoring deployments:

- The **Security Update Status** report and Feature Update Status report provide not only a high-level status (Failed / Progress stalled / Deferred / etc.), but a detailed stage-level representation of where, in the update process, the device was last reported to be, relative to the specific update. By drilling into the detailed status, you may find that a particular category is unusually high and can be further examined for potential remediation actions.

- The **Feature Update Status** report divides devices to provide Deployment Status by Servicing Channel. In addition, Safeguards (also known as Compatibility Holds) uses diagnostic data to determine if an upgrade is not deploying because of third-party compatibility issues with the upgrade known to Microsoft. By default, Microsoft does not allow offering an upgrade until the issue has been resolved and it has been determined that the system can now safely and smoothly install a feature update. (Compatibility holds are optional, and you may opt-out by properly setting a registry key.)

- **Compatibility Holds** – part of the Feature Update Status Report – are generated when the system determines that devices are not able to update due to known issues. It is important to update or remove the driver or app causing the hold so that the feature update can continue. Holds are released over time as diagnostic data is analyzed and fixes are addressed so it is vital to address issues so that feature updates can succeed and monthly servicing updates can be offered.

- The **Need Attention!** report provides a breakdown of all Windows 10 device and update issues detected by Update Compliance. In addition to a count of devices within Failed/Stalled/Canceled/Rollback/Uninstalled categories, a List of Queries section is available with more detailed information that may assist in setting polices to increase velocity. For example:

    - **Deferral configurations for Feature Update**. Have you set a Deadline Policy? Or are you allowing users to defer over and over? (See Deferral Policies in Disabling conflicting or legacy policies.)

    - **Devices pending reboot to complete update**. Are these devices capable of completing an update? (See Compliance deadlines.)

    - **Deferral configurations for Quality Updates**. Is a delayed feature update responsible for a delay in quality updates? Consider pausing your feature update deployment or setting a maximum deferral of 365 days until the quality update is completed. (See Deferral Policies in Disabling conflicting or legacy policies.)

- **Delivery Optimization in Update Compliance** provides you with information about your Delivery Optimization configuration, including the observed bandwidth savings across all devices that used peer-to-peer distribution over the past 28 days.

Note: While Update Compliance is refreshed every twelve hours, the rate at which each type of data is sent from the device and how long it takes to be ready for Update Compliance varies.

Additional diagnostic data

Remember that optimizing for velocity is an iterative process, and by examining the data around failed or slow-to-update devices, you will find sets of devices that are able to update with a single improvement in policies or processes. Data logs are available in Deployment Optimization and while there are pre-packaged reports, it is also possible to query data. Learn more about Log Queries.

Once your policies are set and you are using deployment strategies to increase velocity, you may find the following data points useful in continuing to increase velocity.

- **Understanding current deployment status.** Some devices may have more than one issue impeding their update velocity. We find it is most useful to remediate issues in a specific sequence, as described by the analysis taxonomy described in Monitoring strategies, since some issues mask the existence of other issues. Use these queries to find current issues:

```
WaaSDeploymentStatus
| where DeploymentStatus != "Update completed"
| summarize dcount(ComputerID) by DetailedStatus

WaaSDeploymentStatus
| summarize dcount(ComputerID) by DeploymentError
```

  Here's how these details map to the analysis taxonomy we use at Microsoft:

  - **Low activity** – Often, these devices show as DeploymentStatus "Progress stalled".
  - **Unsuccessful scan** – Devices that are unable to get past DeploymentStatus "Unknown" may be in this category. It is possible that devices with a DetailedStatus of "Cancelled" may also be in this category.
  - **Compatibility** – Devices with a DetailedStatus of "Compatibility hold" map to this category.
  - **Not offered** – Devices that are unable to get past DeploymentStatus "Unknown" may be in this category. Likewise, a device might be Not Offered if it has a DetailedStatus of "Uninstalled" or "Rollback"
  - **Policy** – Devices with a DetailedStatus of "Scheduled in next X days", "Update deferred" or "Update paused" are in this category.
  - **Disk space** – Devices that have a DeploymentStatus of "Failed" with an error code corresponding to a disk error are in this category.
  - **Problems:** All other devices that have DeploymentStatus of "Failed" are in this category.
  - **In progress** – These devices show up with a DeploymentStatus of "In progress".

    - **Waiting for user restart** – Devices with a DetailedStatus of "Reboot required", "Reboot pending", "Reboot initiated", "Commit", "PreInstall task passed", and "Finalize succeeded" are represented in this category.

- **Month-over-month velocity**. We recommend you record changes in month-over-month adoption rate regularly so that you know when something happens to cause an increase or decrease in velocity. For example, you may have tweaked a power policy; did that translate into higher velocity over the previous month? There may have been a holiday, which may have slowed velocity, meaning that no action is necessary.

- **Operating system versions**. You may find that, across your organization, some operating system versions are updating more quickly than others. In fact, you may find that a particular operating system version is dragging down your overall velocity rate. Remember that a slower velocity on a smaller number of devices may look like a high percentage, but this may be misleading as a smaller percentage of a majority of devices are still waiting for updates. In other words, absolute numbers (instead of percentages) in lagging devices may be a good indicator of velocity overall. To drill down on which OS builds are having more issues than others, use this query:

  ```
  WaaSDeploymentStatus
  | where DeploymentStatus != "Update completed"
  | summarize dcount(ComputerID) by DetailedStatus, OSBuild
  ```

- **Server availability**. It could be that a group of devices are pointing to an update server that is offline. Or it could be that devices are pointing to a WSUS server that is no longer in existence. By finding a pattern in failure, this issue will become apparent and a fix can rectify the problem. See Ensuring updates are available for details.

- **Error codes**. While this list is not exhaustive, you will find a list of common Windows error codes that can assist in troubleshooting devices that are failing:

    - Upgrade Error Codes

    - Windows Error Codes

  To find the number of devices per error code, use the following query and see Taking action for more information.

  ```
  WaaSDeploymentStatus
  | where DeploymentStatus == "Failed"
  | summarize dcount(ComputerID) by DeploymentErrorCode
  ```

- **Feature update is getting in the way of quality updates**. Windows will not install a quality update if a feature update is pending. If a quality update has security fixes, it is important that these updates are installed so your network is not at risk. See Taking action if feature update errors are preventing your quality updates from being deployed.

- **Devices stalled for more than 7 days:** Update Compliance provides Deployment Status, and when a device has started the update process but has no activity within 7 days, you'll find a category called

Progress Stalled. Selecting this status will provide more details on the devices that are stalled. Look for issues that can be resolved with policies as described in the Power policies section of this document.

Taking action

Your first step is to use the recommended policies described in this document and see if those adjustments help your devices that are having trouble get upgraded. If devices are reasonably healthy, this should be all you need to do to increase their update velocity. If that does not work, there are a few more steps to try.

Investigating devices that failed with an error code can be a challenge, but often, the error code itself can give you a clue as to what might be causing a problem. Use the Microsoft Error Lookup Tool to identify the error code meaning, as it may suggest the underlying problem.

For feature updates, error codes might be generated by the update components in the process of preparing to run Windows Setup, or they may be returned by Windows Setup itself. If the error comes from Windows Setup, the SetupDiag tool can help you understand what went wrong. SetupDiag is a standalone diagnostic tool that can be used to obtain details about why a Windows 10 upgrade was unsuccessful. SetupDiag works by examining Windows Setup log files and attempting to parse them in order to determine the root cause of a failure to upgrade the device to Windows 10. Starting in Windows 10, version 2004, Windows Setup includes and automatically executes SetupDiag. The output is available on the device, in the registry and as an XML output file.

To transmit this output to a central reporting server, Windows Setup also enables you to customize a feature update by running your own custom action scripts during and after a feature update – for example, executing a script after a feature update failure, with the purpose of launching SetupDiag (for releases prior to Windows 10, version 2004), and gathering and transmitting SetupDiag output. For more information, see Run custom actions during feature update, and learn more about how to resolve Windows 10 upgrade errors.

If you are unable to resolve issues with a particular device, we recommend you judiciously apply network isolation or re-imaging as necessary to satisfy your overall compliance metrics. A device that is prevented from accessing your network assets can be considered compliant when it comes to reporting risk profile from unpatched devices. For more information, see Unhealthy devices.

If you have followed the guidelines in this document, you should see a higher rate of updates with lower fail numbers. You can then continue monitoring the status of upgrades and tweaking your processes to ensure the velocity is as high as possible, based on the priorities of your organization.

Additional resources

- Monitor the Windows 10 deployment with Configuration Manager
- Use Update Compliance
- Log Queries documentation
- Upgrade Error Codes and Windows Error Codes

Deployment strategy

Service management mindset

The industry is going through a digital transformation in how feature updates are deployed, where IT administrators are steadily moving from a project-based approach to a service management model.

In the legacy approach, each feature update deployment was treated as a "project" where you would identify the update, make a business case, and get capital funding. Each deployment had a beginning, a middle and an end, moving forward until the "upgrade project" was completed.

Many enterprises are now treating deployments as a continual process of updates, rolling out across the organization in waves, and this living service management model requires a new mindset and approach for IT administrators.

To best support this service management model, the update mechanisms in Windows have been enhanced to understand your priorities – and your users' experience – in order to optimize for the lowest impact on your enterprise.

With a service management approach, the process is configured, and instead of "building a release" (e.g. an image), an update is plugged into the process and runs while you monitor for anomalies, spikes in errors, or user impact, and respond to any issues that arise – all without interrupting the entire process.

Some of the benefits of a service management approach include:

- Shorter deployment times
- Better validation
- Lower risk
- Reduction in cost
- Happier users

Service management allows IT administrators to configure their update process based on priorities – security, productivity, cost drivers – and monitor their systems on an ongoing basis, adjusting as needed. By treating deployment as a continuous process instead of discrete events, we are able to build in efficiencies and quality measures that are not available in a project-based system.

To illustrate, when some enterprises were preparing for deployment of a feature update in a project-based approach, the pre-deployment readiness preparation was a fixed set of steps that the IT administrators would set up, build, and run from start-to-finish. In other words, every feature update was an event. And if there was an issue, the deployment came to a halt until the issue was resolved, and then everything started over again. Oftentimes, during the feature update process, a quality update may have been released and blocked because the feature update was not fully deployed, leaving many devices within an organization vulnerable for quite some time – even those devices that otherwise would not have encountered the issue. Those insecure devices, in turn, put the larger organization at risk.

With a service-based approach, policies are set, and tools are used to optimize updates based on set criteria. That criteria are then monitored and if issues arise, the process can continue to move forward or backward for a subset of devices, based on actual diagnostic data. Just because some devices are not able to take a feature update does not mean that the entire process must come to a halt.

Pre-deployment readiness preparation

Recommended: Service-based approach	Project-based approach

• Telemetry-based management
• Analytics suite, leveraged for monitoring
• Flighting to ensure app readiness
• Hardware compatibility insights from Microsoft
• Peer training
	• OS readiness
	• Network readiness
	• Application compatibility
	  ○ Internal application readiness
	  ○ External application readiness
	• Driver update policy
	• Whole user base training

A service management approach includes repeatable processes that are defined once (or at least infrequently, with periodic revisions) and then run repeatedly. For example, a "targeted" ring might be defined with representative devices and software across the organization so that with each rollout, it is no longer necessary to have a task called "app compatibility," as this compatibility check is handled automatically with reports and diagnostic data from flighting devices, freeing up your time to focus only on issues if they arise. At minimum, you can limit the app compatibility testing effort to just the most critical applications in your organization.

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. And once this process is put into place for feature updates, quality updates become a lightweight procedure that is simple and fast to execute, ultimately increasing velocity.

What follows are best practices for service management deployment strategies that can further optimize velocity as well as increase satisfaction for your users during the Windows 10 update experience.

Ring deployment

Increasing velocity means deploying updates in the most efficient way possible while considering the productivity of your workforce.

In moving to a service management model, you need effective ways of rolling out updates to representative groups of devices, and we've found that ring-based deployment is a methodology that works well for us at Microsoft as well as many other organizations across the globe.

Deployment rings in Windows 10 are similar to the deployment groups most organizations constructed for previous major revision upgrades – they are simply a method by which to separate machines into a deployment timeline.

At the highest level, each "ring" constitutes a group of users or devices that receive a particular update concurrently. IT administrators set time-deferral or adoption-completion criteria for each ring before deploying to the next broader ring of devices or users.

A common ring structure consists of three rings:

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

Note: Organizations often call their "rings" by different names, such as:

- First > Fast > Broad
- Canaries > Early Adopters > Users
- Preview > Broad > Critical

How many rings should I have?

There are no hard-and-fast rules on exactly how many rings to have for your deployments. As mentioned above, for mission-critical devices, you may want to have a separate ring to ensure zero downtime. If you have a large organization, you may want to consider geo-distribution and/or size of rings so that helpdesk resources are available, from call quantity and time zone perspectives, in case of incidents. Consider the needs of your business and introduce rings that make sense for your organization.

Ring advancement

There are two-types of strategies for how deployments progress from one ring to the next – one follows the service-based mindset; the other, project-based.

- **Red button (service-based)**. Assumes that content is good until proven bad. Content flows until an issue is discovered, and 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.

When it comes to deployments, having manual steps in the process usually impedes velocity, thus a Red Button strategy is preferred.

Ring deployment resources

- Microsoft Ignite: Strategic and tactical considerations for ring-based Windows 10 deployments
- Build deployment rings for Windows 10 updates
- How to progressively expose your Azure DevOps extension releases in production to validate, before impacting all users

Preview ring

The purpose of the Preview ring is to evaluate the new features of the update. This is specifically *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 is the time the design and planning work happens so that when the public update is actually shipped, you can have greater confidence in the update.

Note: If you are not part of the Windows Insider Program, you should be. 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 will not lose productivity if something goes wrong. In general, these are IT pros, and perhaps a few people in the business organization.

What happens during this plan and prepare period?

- 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 will benefit.
- Understand why you are putting the update out.
- Plan for usage feedback.

Remember, you are working with pre-release software in the Preview ring and you will be evaluating features and testing the build for a targeted release.

> **Important:** If you are using Windows Insider (pre-release) builds for your preview ring and you are using WSUS or Windows Update for Business, 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**

For more information on Preview Rings, see Windows Insider Program for Business.

Limited ring

The purpose of the Limited ring is to validate the build on representative devices across the network. During this period, data and feedback is 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 assist in monitoring the deployment. Learn more in Ring deployment planning.

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, and it is key that the people selected for this ring are using their devices regularly in order to generate the data you will need to make a decision for broader deployment across your organization. The IT department, lab machines, and users with the most cutting-edge hardware usually don't have the applications or drivers of devices that are truly a representative sample of your network.

What happens during the pilot and validate period?

- 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 (See Monitoring and enforcement.), quickly addressing any issues that may arise. Ensure you monitor for sufficient adoption within this ring, because your Limited ring represents your organization across the board, and when you achieve sufficient adoption, you can have confidence that your broader deployment will run more smoothly – ultimately accelerating the overall adoption rate for updates.

Broad deployment

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

Who goes in the Broad deployment ring?

> **Note:** In some instances, you may hold back on mission critical devices (e.g. medical devices) until the broad deployment ring is complete. Get best practices and recommendations for deploying Windows 10 Feature updates to mission critical devices.

In most businesses, the Broad ring includes the rest of your organization, and because you've done the work in the previous ring to vet stability and minimize disruption – and have diagnostic data to support your decision such as help desk data or those detailed in Monitoring and enforcement – broad deployment can roll out relatively quickly.

What happens during the Broad deployment period?

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

Ring deployment planning

While, in the past, we have provided methods for analyzing your deployments, these have generally been stand-alone tools to assess, manage and execute deployments. In other words, you would generate an analysis, make a deployment strategy, and then move to your console for implementation, repeating these steps this for each deployment. We have combined many of these tasks, and more, into a single interface with Desktop Analytics.

Desktop Analytics is a cloud-based service and a key component of Microsoft Endpoint Manager). Using artificial intelligence and machine learning, Desktop Analytics is a powerful tool to give you insights and intelligence to make informed decisions about the readiness of your Windows clients.

Reminder: Desktop Analytics is intended for use primarily with Configuration Manager.

In Windows 10 deployments, we have seen compatibility issues on <0.5% of apps when using Desktop Analytics.

Using Desktop Analytics with Microsoft Endpoint Manager can help you assess app compatibility with the latest feature update and create groups that represent the broadest number of hardware and software configurations on the smallest set of devices across your organization. In addition, Desktop Analytics can provide you with a device and software inventory and identify issues, giving you data that equate to actionable decisions.

Important: Desktop Analytics does not support preview (Windows Insider) builds; use Configuration Manager to deploy to your Preview ring. As noted above, the Preview ring is a small group of devices that highly represents your ecosystem in terms of app, driver, and hardware diversity.

Deployment plan options

There are two ways to implement a ring deployment plan, depending on how you manage your devices:

- **If you are using Configuration Manager**: Desktop Analytics provides end-to-end deployment plan integration so that you can also kick off phased deployments within a ring. Learn more about deployment plans in Desktop Analytics.
- **If you are using Microsoft Intune**: Create deployment plans directly in Intune.

Desktop Analytics resources

- How to set up Desktop Analytics
- Tutorial: Deploy Windows 10 to Pilot
- Desktop Analytics Documentation
- Intune deployment planning, design, and implementation guide

Microsoft

Policy and settings reference guide

The following settings are recommended to increase update velocity while taking into account the user's experience. You can set these either using an MDM service such as Intune, or through Group Policy. Group Policy paths shown are relative to the path **ComputerConfiguration/ AdministrativeTemplates**/.

Velocity and compliance

Policy name	Recommended settings	Notes
MDM: Update/ConfigureDeadlineForQualityUpdates **GP:** WindowsComponents/WindowsUpdate/ ConfigureDeadlineForQualityUpdates	3	By specifying deadlines for updates and reboots, deployment will happen in the background and users will be appropriately notified when restarts are required.
MDM: Update/ConfigureDeadlineForFeatureUpdates **GP:** WindowsComponents/WindowsUpdate/ ConfigureDeadlineForFeatureUpdates	7	By specifying deadlines for updates and reboots, deployment will happen in the background and users will be appropriately notified when restarts are required.
MDM: Update/ConfigureDeadlineGracePeriod **GP:** WindowsComponents/WindowsUpdate/ ConfigureDeadlineGracePeriod	2	Especially useful in cases where a user has been away for many days (e.g. on vacation) so that the device will not be forced into an immediate update upon the user's return.
MDM: Update/ConfigureDeadlineNoAutoReboot **GP:** WindowsComponents/WindowsUpdate/ ConfigureDeadlineNoAutoReboot	0 - Disabled	Windows has heuristics based on user interactions that dynamically identifies the least disruptive time for automatic restart.
DEFERRALS & PAUSE		
MDM: Update/DeferFeatureUpdatesPeriodInDays **GP:** WindowsComponents/WindowsUpdate/ DeferFeatureUpdates	0	Fastest velocity is no deferral. You may extend this to allow for greater test times or to

Policy name	Recommended settings	Notes
		stage deployment across your enterprise. If you are considering using deferral of feature updates because they take precedence over quality update and you need to maximize quality update velocity, you should use pause of feature updates instead. Deferral on a device that is more than one feature update behind may simply switch to installing the next older update if you choose to defer. To validate earlier, try using Windows Insider Program for Business to test feature updates in your environment before they are released.
MDM: Update/DeferQualityUpdatesPeriodInDays **GP:** WindowsComponents/WindowsUpdate/ DeferQualityUpdates	0	Defer 2-3 days only if you want to test first with a smaller ring of devices before installing these deferred devices.
MDM: Updates/PauseFeatureUpdatesStartTime **GP:** WindowsComponents/WindowsUpdate/ PauseFeatureUpdatesStartId	Not configured	Do not pause unless you encounter a known issue where you want to allow time for resolution.
MDM: Updates/PauseQualityUpdatesStartTime **GP:** WindowsComponents/WindowsUpdate/ PauseQualityUpdatesStartId	Not configured	Do not pause unless you encounter a known issue where you want to allow time for resolution.
MDM: Update/SetDisablePauseUXAccess **GP:** WindowsComponents/WindowsUpdate/ SetDisablePauseUXAccess	1	Prevents users from pausing quality and feature updates.
RESTART BEHAVIOR		
MDM: Update/NoAutoRebootWithLoggedOnUsers	0	Prevents a user that is logged on from stopping an automatic reboot when Windows finds a

Policy name	Recommended settings	Notes
GP: WindowsComponents/WindowsUpdate/ NoAutoRebootWithLoggedOnUsers		good time to update when the user will not be impacted.
MDM: Update/UpdateNotificationLevel **GP:** WindowsComponents/WindowsUpdate/ UpdateNotificationLevel	0	Prevents notifications from being disabled. If the device is serving as a digital sign or kiosk, you should set this to 2.
MDM: Update/AllowAutoUpdates **GP:** WindowsComponents/WindowsUpdate/ AutoUpdateMode	Not configured	It is best to allow Intelligent Active Hours control the behavior here. If you must set this policy, set it to 4 for GP, or 6 (Reset) for MDM.

Accounting for low activity devices

Policy name	Recommended settings	Notes
ACTIVE HOURS		
MDM: Update/ActiveHoursMaxRange **GP**: WindowsComponents/WindowsUpdate/ ActiveHoursMaxRange	Not configured	Allows Windows to intelligently adapt active hours based on the user's behavior.
MDM: Update/ActiveHoursStart **GP**: WindowsComponents/WindowsUpdate/ ActiveHoursStartTime	Not configured	Allows Windows to intelligently adapt active hours based on the user's behavior.
MDM: Update/ActiveHoursEnd **GP**: WindowsComponents/WindowsUpdate/ ActiveHoursEndTime	Not configured	Allows Windows to intelligently adapt active hours based on the user's behavior.
MDM: Update/ScheduledInstallTime **GP**: WindowsComponents/Windows Update/ AutoUpdateSchTime	Not configured	Do not set as you may conflict with Active Hours
POWER		
MDM: Power/EnergySaverBatteryThresholdOnBattery **GP**: System/PowerManagement/EnergySaverSettings/ EnterEsBattThresholdDC	40%	Assumes modern device with strong battery.

Policy name	Recommended settings	Notes
MDM: Power/EnergySaverBatteryThresholdPluggedIn **GP**: System/PowerManagement/EnergySaverSettings/ EnterEsBattThresholdDC	40%	Assumes modern device with strong battery.
MDM: Power/AllowStandbyStatesWhenSleepingOnBattery **GP**: System/PowerManagement/SleepSettings/ AllowStandbyStatesDC_2	Enabled or Not configured	Allows device to sleep, enabling the device to wake up for updates.
MDM: Power/AllowStandbyWhenSleepingPluggedIn **GP**: System/PowerManagement/SleepSettings/ AllowStandbyStatesAC_2	Enabled or Not configured	Allows device to sleep, enabling the device to wake up for updates.
MDM: Power/SelectLidCloseActionOnBattery **GP**: System/PowerManagement/ButtonSettings/ DCSystemLidAction_2	1 (Sleep)	Allows device to sleep, enabling the device to wake up for updates.
MDM: Power/SelectLidCloseActionPluggedIn **GP**: System/PowerManagement/ButtonSettings/ ACSystemLidAction_2	1 (Sleep)	Allows device to sleep, enabling the device to wake up for updates.
MDM: Power/HibernateTimeoutOnBattery **GP**: System/PowerManagement/SleepSettings/ DCHibernateTimeOut_2	Not configured	Prevents device from Hibernate, allowing device to take an update.
MDM: Power/HibernateTimeoutPluggedIn **GP**: System/PowerManagement/SleepSettings/ ACHibernateTimeOut_2	Not configured	Prevents device from Hibernate, allowing device to take an update.
SCANNING		
MDM: Update/AllowUpdateService **GP**: WindowsComponents/WindowsUpdate/ AllowUpdateService	1	Allows device to scan Windows Update for updates
MDM: Update/AllowMUUpdateService **GP**: WindowsComponents/WindowsUpdate/ AllowMUUpdateService	1	For best compliance, we recommend you enable Microsoft Update to get the latest in updates for Microsoft software as well as third-party drivers.
MDM: Update/ DisableAllUnmanagedUpdateServicesWhenWSUS **GP**: WindowsComponents/WindowsUpdate/ DoNotConnectToWindowsUpdateInternetLocations	0	Allows device to scan Windows Update for OS and store updates

Policy name	Recommended settings	Notes
MDM: Update/SetDisableUXWUAccess **GP**: WindowsUpdate/SetDisableUXWUAccess	0	Allows users to kick off a scan for updates
MDM: Update/DisableDualScan **GP**: WindowsComponents/WindowsUpdate/DisableDualScan	0	Allows device to scan for OS updates from Windows Update.
MDM: Update/ExcludeWUDriversInQualityUpdate **GP**: WindowsComponents/WindowsUpdate/ExcludeWUDriversInQualityUpdate	0	Allows device to scan for everything
MDM: Update/DetectionFrequency **GP**: WindowsComponents/WindowsUpdate/DetectionFrequency	6	Allows the device to scan reasonably often (every 6 hours)

Deployment policies

Policy name	Recommended settings	Notes
DELIVERY OPTIMIZATION		
MDM: DeliveryOptimization/DOGroupId **GP**: WindowsComponents/DeliveryOptimization/GroupId	Not configured	Configuration Manager will set this policy to create peer groups dynamically. If you are not using Configuration Manager, then you can set a GUID to define a custom peer group.
MDM: DelieryOptimization/DODownloadMode **GP**: WindowsComponents/DeliveryOptimization/DownloadMode	2	Use in combination with Group ID or "Select the Source of Group ID" to enable group peering blended with HTTP.
MDM: DeliveryOptimization/DOMinFileSizeToCache **GP**: WindowsComponents/DeliveryOptimization/MinFileSizeToCache	10	Specifies the minimum content file size in MB enabled to use peer caching.
MDM: DeliveryOptimization/DOMinBatteryPercentageAllowedToUpload **GP**: WindowsComponents/DeliveryOptimization/MinBatteryPercentageAllowedToUpload	60	Allows uploads on battery power. For organizations with high-end devices.

Policy name	Recommended settings	Notes
MDM: DeliveryOptimization/DOMaxCacheAge **GP**: DeliveryOptimization/MaxCacheAge	604800 (7 days)	Best for labs with high storage and AC-powered devices.
MDM: DeliveryOptimization/ DODelayBackgroundDownloadFromHttp **GP**: WindowsComponents/DeliveryOptimization/ DelayBackgroundDownloadFromHttp	300	To increase the usage of peers in favor of HTTP in background downloads.
MDM: DeliveryOptimization/ DODelayForegroundDownloadFromHttp **GP**: WindowsComponents/DeliveryOptimization/ DelayForegroundDownloadFromHttp	60	To increase the usage of peers in favor of HTTP in foreground downloads.
ADDITIONAL DEPLOYMENT POLICIES		
MDM: Update/ManagePreviewBuilds **GP**: WindowsComponents/WindowsUpdate/ ManagePreviewBuildsId	Not configured	If device is in preview ring: 2 – Enable Preview Builds
MDM: Update/BranchReadinessLevel **GP**: WindowsComponents/WindowsUpdate/ BranchReadinessLevelId	16 – Semi-Annual Channel Releases	Selects when preview builds and feature updates are received. If the device is in a preview ring, set to: 4 -Windows Insider Program Slow
MDM: Update/AllowAutoWindowsUpdateDownload OverMeteredNetwork **GP**: WindowsComponents/WindowsUpdate/ AllowAutoWindowsUpdate DownloadOverMeteredNetwork	1	If all possible metered connections are known safe and okay, set to 1 to maximize velocity. Otherwise, leave the value at: Default – 0, do not allow updates to be automatically downloaded over a metered network

Policies to disable or not configure

The following policies should be set to **Disabled** or **Not configured** as they may conflict with policies set above; these policies are outdated and/or in the process of being deprecated.

MDM policy name	GP policy name
Update/AutomaticMaintenanceWakeUp	WindowsComponents/WindowsUpdate/AutoUpdateMode
Update/AutoRestartDeadlinePeriodInDays	WindowsComponents/WindowsUpdate/AutoRestartDeadline
Update/AutoRestartDeadlinePeriodInDaysForFeatureUpdates	WindowsComponents/WindowsUpdate/AutoRestartDeadlineForFeatureUpdates
Update/AutoRestartNotificationSchedule	WindowsComponents/WindowsUpdate/AutoRestartNotificationSchd
Update/AutoRestartRequiredNotificationDismissal	WindowsComponents/WindowsUpdate/AutoRestartNotificationDismissal
Update/DeferUpdatePeriod	WindowsComponents/WindowsUpdate/DeferUpdatePeriodId
Update/DeferUpgradePeriod	WindowsComponents/WindowsUpdate/DeferUpgradePeriodId
Update/EngagedRestartDeadline	WindowsComponents/WindowsUpdate/EngagedRestartDeadline
Update/EngagedRestartDeadlineForFeatureUpdates	WindowsComponents/WindowsUpdate/EngagedRestartDeadlineForFeatureUpdates
Update/EngagedRestartSnoozeSchedule	WindowsComponents/WindowsUpdate/EngagedRestartSnoozeSchedule
Update/EngagedRestartSnoozeScheduleForFeatureUpdates	WindowsComponents/WindowsUpdate/EngagedRestartSnoozeScheduleForFeatureUpdates
Update/EngagedRestartTransitionSchedule	WindowsComponents/WindowsUpdate/EngagedRestartTransitionSchedule
Update/EngagedRestartTransitionScheduleForFeatureUpdates	WindowsComponents/WindowsUpdate/EngagedRestartTransitionScheduleForFeatureUpdates
Update/PauseDeferrals	WindowsComponents/WindowsUpdate/PauseDeferralsId
Update/PauseFeatureUpdates	WindowsComponents/WindowsUpdate/PauseFeatureUpdatesId
Update/PauseQualityUpdates	WindowsComponents/WindowsUpdate/PauseQualityUpdatesId

MDM policy name	GP policy name
Update/PhoneUpdateRestrictions	WindowsComponents/WindowsUpdate/ PhoneUpdateRestrictionsId
Update/RequireDeferUpgrade	WindowsComponents/WindowsUpdate/ DeferUpgradePeriodId
Update/RequireUpdateApproval	WindowsComponents/WindowsUpdate/ RequireUpdateApproval
Update/ScheduledInstallDay	WindowsComponents/WindowsUpdate/ AutoUpdateSchDay
Update/ScheduledInstallEveryWeek	WindowsComponents/WindowsUpdate/ AutoUpdateSchEveryWeek
Update/ScheduledInstallFirstWeek	WindowsComponents/WindowsUpdate/ AutoUpdateSchFirstWeek
Update/ScheduledInstallFourthWeek	WindowsComponents/WindowsUpdate/ ScheduledInstallFourthWeek
Update/ScheduledInstallSecondWeek	WindowsComponents/WindowsUpdate/ ScheduledInstallSecondWeek
Update/ScheduledInstallThirdWeek	WindowsComponents/WindowsUpdate/ ScheduledInstallThirdWeek
Update/ScheduleImminentRestartWarning	WindowsComponents/WindowsUpdate/ RestartWarn
Update/ScheduleRestartWarning	WindowsComponents/WindowsUpdate/ RestartWarnRemind
Update/SetAutoRestartNotificationDisable	WindowsComponents/WindowsUpdate/ AutoRestartNotificationDisable