

# Contents

## Privacy

Beginning your General Data Protection Regulation (GDPR) journey for Windows 10

Windows 10 and the GDPR for IT Decision Makers

Windows 10 personal data services configuration

Configure Windows diagnostic data in your organization

Diagnostic Data Viewer Overview

Basic level Windows diagnostic data events and fields

Windows 10, version 1803 basic level Windows diagnostic events and fields

Windows 10, version 1709 basic level Windows diagnostic events and fields

Windows 10, version 1703 basic level Windows diagnostic events and fields

Enhanced level Windows diagnostic data events and fields

Windows 10, version 1709 enhanced diagnostic data events and fields used by Windows Analytics

Full level categories

Windows 10, version 1709 and newer diagnostic data for the Full level

Windows 10, version 1703 diagnostic data for the Full level

Manage Windows 10 connection endpoints

Windows 10, version 1709, connection endpoints for non-Enterprise editions

Windows 10, version 1803, connection endpoints for non-Enterprise editions

Manage connections from Windows operating system components to Microsoft services

# Beginning your General Data Protection Regulation (GDPR) journey for Windows 10

5/10/2018 • 26 minutes to read • [Edit Online](#)

This article provides info about the GDPR, including what it is, and the products Microsoft provides to help you to become compliant.

## Introduction

On May 25, 2018, a European privacy law is due to take effect that sets a new global bar for privacy rights, security, and compliance.

The General Data Protection Regulation, or GDPR, is fundamentally about protecting and enabling the privacy rights of individuals. The GDPR establishes strict global privacy requirements governing how you manage and protect personal data while respecting individual choice — no matter where data is sent, processed, or stored.

Microsoft and our customers are now on a journey to achieve the privacy goals of the GDPR. At Microsoft, we believe privacy is a fundamental right, and we believe that the GDPR is an important step forward for clarifying and enabling individual privacy rights. But we also recognize that the GDPR will require significant changes by organizations all over the world.

We have outlined our commitment to the GDPR and how we are supporting our customers within the [Get GDPR compliant with the Microsoft Cloud](#) blog post by our Chief Privacy Officer [Brendon Lynch](#) and the [Earning your trust with contractual commitments to the General Data Protection Regulation](#) blog post by [Rich Sauer](#) - Microsoft Corporate Vice President & Deputy General Counsel.

Although your journey to GDPR-compliance may seem challenging, we're here to help you. For specific information about the GDPR, our commitments and how to begin your journey, please visit the [GDPR section of the Microsoft Trust Center](#).

## GDPR and its implications

The GDPR is a complex regulation that may require significant changes in how you gather, use and manage personal data. Microsoft has a long history of helping our customers comply with complex regulations, and when it comes to preparing for the GDPR, we are your partner on this journey.

The GDPR imposes rules on organizations that offer goods and services to people in the European Union (EU), or that collect and analyze data tied to EU residents, no matter where those businesses are located. Among the key elements of the GDPR are the following:

- **Enhanced personal privacy rights.** Strengthened data protection for residents of EU by ensuring they have the right to access to their personal data, to correct inaccuracies in that data, to erase that data, to object to processing of their personal data, and to move it.
- **Increased duty for protecting personal data.** Reinforced accountability of organizations that process personal data, providing increased clarity of responsibility in ensuring compliance.
- **Mandatory personal data breach reporting.** Organizations that control personal data are required to report personal data breaches that pose a risk to the rights and freedoms of individuals to their supervisory authorities without undue delay, and, where feasible, no later than 72 hours once they become aware of the breach.

As you might anticipate, the GDPR can have a significant impact on your business, potentially requiring you to update privacy policies, implement and strengthen data protection controls and breach notification procedures, deploy highly transparent policies, and further invest in IT and training. Microsoft Windows 10 can help you effectively and efficiently address some of these requirements.

## Personal and sensitive data

As part of your effort to comply with the GDPR, you will need to understand how the regulation defines personal and sensitive data and how those definitions relate to data held by your organization.

The GDPR considers personal data to be any information related to an identified or identifiable natural person. That can include both direct identification (such as, your legal name) and indirect identification (such as, specific information that makes it clear it is you the data references). The GDPR also makes clear that the concept of personal data includes online identifiers (such as, IP addresses, mobile device IDs) and location data.

The GDPR introduces specific definitions for genetic data (such as, an individual's gene sequence) and biometric data. Genetic data and biometric data along with other sub categories of personal data (personal data revealing racial or ethnic origin, political opinions, religious or philosophical beliefs, or trade union membership; data concerning health; or data concerning a person's sex life or sexual orientation) are treated as sensitive personal data under the GDPR. Sensitive personal data is afforded enhanced protections and generally requires an individual's explicit consent where these data are to be processed.

### Examples of info relating to an identified or identifiable natural person (data subject)

This list provides examples of several types of info that will be regulated through GDPR. This is not an exhaustive list.

- Name
- Identification number (such as, SSN)
- Location data (such as, home address)
- Online identifier (such as, e-mail address, screen names, IP address, device IDs)
- Pseudonymous data (such as, using a key to identify individuals)
- Genetic data (such as, biological samples from an individual)
- Biometric data (such as, fingerprints, facial recognition)

## Getting started on the journey towards GDPR compliance

Given how much is involved to become GDPR-compliant, we strongly recommend that you don't wait to prepare until enforcement begins. You should review your privacy and data management practices now. We recommend that you begin your journey to GDPR compliance by focusing on four key steps:

- **Discover.** Identify what personal data you have and where it resides.
- **Manage.** Govern how personal data is used and accessed.
- **Protect.** Establish security controls to prevent, detect, and respond to vulnerabilities and data breaches.
- **Report.** Act on data requests, report data breaches, and keep required documentation.



For each of the steps, we've outlined example tools, resources, and features in various Microsoft solutions, which can be used to help you address the requirements of that step. While this article isn't a comprehensive "how to," we've included links for you to find out more details, and more information is available in the [GDPR section of the Microsoft Trust Center](#).

## Windows 10 security and privacy

As you work to comply with the GDPR, understanding the role of your desktop and laptop client machines in creating, accessing, processing, storing and managing data that may qualify as personal and potentially sensitive data under the GDPR is important. Windows 10 provides capabilities that will help you comply with the GDPR requirements to implement appropriate technical and organizational security measures to protect personal data.

With Windows 10, your ability to protect, detect and defend against the types of attacks that can lead to data breaches is greatly improved. Given the stringent requirements around breach notification within the GDPR, ensuring that your desktop and laptop systems are well defended will lower the risks you face that could result in costly breach analysis and notification.

In this section, we'll talk about how Windows 10 provides capabilities that fit squarely in the **Protect** stage of your journey, including these 4 scenarios:

- **Threat protection: Pre-breach threat resistance.** Disrupt the malware and hacking industry by moving the playing field to one where they lose the attack vectors that they depend on.
- **Threat protection: Post-breach detection and response.** Detect, investigate, and respond to advanced threats and data breaches on your networks.
- **Identity protection.** Next generation technology to help protect your user's identities from abuse.
- **Information protection.** Comprehensive data protection while meeting compliance requirements and maintaining user productivity.

These capabilities, discussed in more detail below with references to specific GDPR requirements, are built on top of advanced device protection that maintains the integrity and security of the operating system and data.

A key provision within the GDPR is data protection by design and by default, and helping with your ability to meet this provision are features within Windows 10 such as the Trusted Platform Module (TPM) technology designed to provide hardware-based, security-related functions. A TPM chip is a secure crypto-processor that is designed to carry out cryptographic operations.

The chip includes multiple physical security mechanisms to make it tamper resistant, and malicious software is unable to tamper with the security functions of the TPM. Some of the key advantages of using TPM technology are that you can:

- Generate, store, and limit the use of cryptographic keys.
- Use TPM technology for platform device authentication by using the TPM's unique RSA key, which is

burned into itself.

- Help to ensure platform integrity by taking and storing security measurements.

Additional advanced device protection relevant to your operating without data breaches include Windows Trusted Boot to help maintain the integrity of the system by ensuring malware is unable to start before system defenses.

### **Threat protection: Pre-breach threat resistance**

The GDPR requires you to implement appropriate technical and organizational security measures to protect personal data.

Your ability to meet this requirement to implement appropriate technical security measures should reflect the threats you face in today's increasingly hostile IT environment. Today's security threat landscape is one of aggressive and tenacious threats. In previous years, malicious attackers mostly focused on gaining community recognition through their attacks or the thrill of temporarily taking a system offline. Since then, attacker's motives have shifted toward making money, including holding devices and data hostage until the owner pays the demanded ransom.

Modern attacks increasingly focus on large-scale intellectual property theft; targeted system degradation that can result in financial loss; and now even cyberterrorism that threatens the security of individuals, businesses, and national interests all over the world. These attackers are typically highly trained individuals and security experts, some of whom are in the employ of nation states that have large budgets and seemingly unlimited human resources. Threats like these require an approach that can meet this challenge.

Not only are these threats a risk to your ability to maintain control of any personal or sensitive data you may have, but they are a material risk to your overall business as well. Consider recent data from Ponemon Institute, Verizon, and Microsoft:

- The average cost of the type of data breach the GDPR will expect you to report is \$3.5M. (Ponemon Institute).
- 63% of these breaches involve weak or stolen passwords that the GDPR expects you to address. (2016 Data Breach Investigations Report, Verizon Enterprise).
- Over 300,000 new malware samples are created and spread every day making your task to address data protection even more challenging. (Microsoft Malware Protection Center, Microsoft).

As seen with recent ransomware attacks, once called the "black plague" of the Internet, attackers are going after bigger targets that can afford to pay more, with potentially catastrophic consequences. Desktops and laptops, that contain personal and sensitive data, are commonly targeted where control over data might be lost.

In response to these threats and as a part of your mechanisms to resist these types of breaches so that you remain in compliance with the GDPR, Windows 10 provides built in technology, detailed below including the following:

- Windows Defender Antivirus to respond to emerging threats on data.
- Microsoft Edge to systemically disrupt phishing, malware, and hacking attacks.
- Windows Defender Device Guard to block all unwanted applications on client machines.

### **Responding to emerging data threats**

Windows Defender Antivirus is a built-in antimalware solution that provides security and antimalware management for desktops, portable computers, and servers. In Windows 10, it uses a multi-pronged approach to improve antimalware:

- **Cloud-delivered protection.** Helps to detect and block new malware within seconds, even if the malware has never been seen before.
- **Rich local context.** Improves how malware is identified. Windows 10 informs Windows Defender

Antivirus not only about content like files and processes, but also where the content came from, where it's been stored, and more.

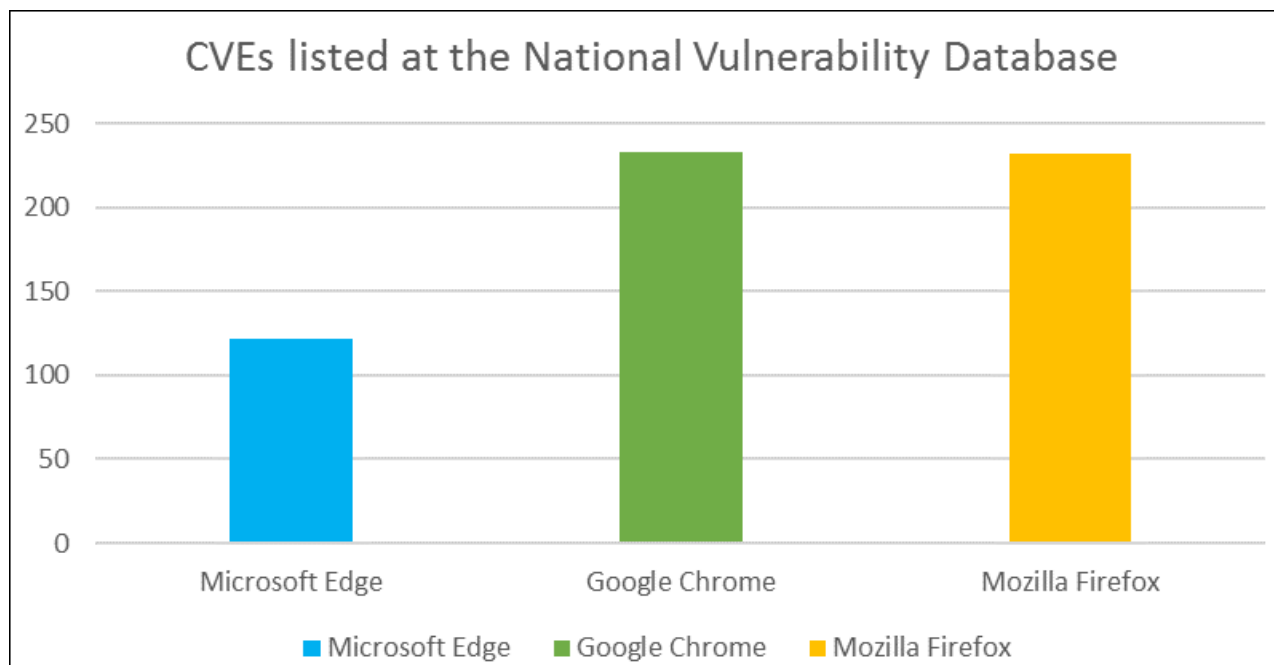
- **Extensive global sensors.** Help to keep Windows Defender Antivirus current and aware of even the newest malware. This is accomplished in two ways: by collecting the rich local context data from end points and by centrally analyzing that data.
- **Tamper proofing.** Helps to guard Windows Defender Antivirus itself against malware attacks. For example, Windows Defender Antivirus uses Protected Processes, which prevents untrusted processes from attempting to tamper with Windows Defender Antivirus components, its registry keys, and so on.
- **Enterprise-level features.** Give IT pros the tools and configuration options necessary to make Windows Defender Antivirus an enterprise-class antimalware solution.

#### Systemically disrupting phishing, malware, and hacking attacks

In today's threat landscape, your ability to provide those mechanisms should be tied to the specific data-focused attacks you face through phishing, malware and hacking due to the browser-related attacks.

As part of Windows 10, Microsoft has brought you Microsoft Edge, our safest and most secure browser to-date. Over the past two years, we have been continuously innovating, and we're proud of the progress we've made. This quality of engineering is reflected by the reduction of Common Vulnerabilities and Exposures (CVE) when comparing Microsoft Edge with Internet Explorer over the past year. Browser-related attacks on personal and sensitive data that you will need to protect under the GDPR means this innovation in Windows 10 is important.

While no modern browser — or any complex application — is free of vulnerabilities, many of the vulnerabilities for Microsoft Edge have been responsibly reported by professional security researchers who work with the Microsoft Security Response Center (MSRC) and the Microsoft Edge team to ensure customers are protected well before any attacker might use these vulnerabilities in the wild. Even better, there is no evidence that any vulnerabilities have been exploited in the wild as zero-day attacks.



However, many businesses worldwide have come under increasing threat of targeted attacks, where attackers are crafting specialized attacks against a specific business, attempting to take control of corporate networks and data.

#### Blocking all unwanted apps

Application Control is your best defense in a world where there are more than 300,000 new malware samples each day. As part of Windows 10, Windows Defender Device Guard is a combination of enterprise-related hardware and software security features that, when configured together, will lock a device down so that it can only run trusted applications that you define in your code integrity policies. If the app isn't trusted it can't run, period.

With hardware that meets basic requirements, it also means that even if an attacker manages to get control of the Windows kernel, he or she will be much less likely to be able to run malicious executable code. With appropriate hardware, Windows Defender Device Guard can use the new virtualization-based security in Windows 10 to isolate the Code Integrity service from the Microsoft Windows kernel itself. In this case, the Code Integrity service runs alongside the kernel in a Windows hypervisor-protected container.

Windows Defender Device Guard protects threats that can expose personal or sensitive data to attack, including:

- Exposure to new malware, for which the "signature" is not yet known
- Exposure to unsigned code (most malware is unsigned)
- Malware that gains access to the kernel and then, from within the kernel, captures sensitive information or damages the system
- DMA-based attacks, for example, attacks launched from a malicious device that read secrets from memory, making the enterprise more vulnerable to attack; and
- Exposure to boot kits or to a physically present attacker at boot time.

### Threat protection: Post-breach detection and response

The GDPR includes explicit requirements for breach notification where a personal data breach means, "a breach of security leading to the accidental or unlawful destruction, loss, alteration, unauthorized disclosure of, or access to, personal data transmitted, stored or otherwise processed."

As noted in the Windows Security Center white paper, [Post Breach: Dealing with Advanced Threats](#), "Unlike pre-breach, post-breach assumes a breach has already occurred – acting as a flight recorder and Crime Scene Investigator (CSI). Post-breach provides security teams the information and toolset needed to identify, investigate, and respond to attacks that otherwise will stay undetected and below the radar."

### Insightful security diagnostic data

For nearly two decades, Microsoft has been turning threats into useful intelligence that can help fortify our platform and protect customers. Today, with the immense computing advantages afforded by the cloud, we are finding new ways to use our rich analytics engines driven by threat intelligence to protect our customers.

By applying a combination of automated and manual processes, machine learning and human experts, we can create an Intelligent Security Graph that learns from itself and evolves in real-time, reducing our collective time to detect and respond to new incidents across our products.



The scope of Microsoft's threat intelligence spans, literally, billions of data points: 35 billion messages scanned monthly, 1 billion customers across enterprise and consumer segments accessing 200+ cloud services, and 14 billion authentications performed daily. All this data is pulled together on your behalf by Microsoft to create the Intelligent Security Graph that can help you protect your front door dynamically to stay secure, remain productive, and meet the requirements of the GDPR.

### Detecting attacks and forensic investigation

Even the best endpoint defenses may be breached eventually, as cyberattacks become more sophisticated and targeted.

Windows Defender Advanced Threat Protection (ATP) helps you detect, investigate, and respond to advanced attacks and data breaches on your networks. GDPR expects you to protect against attacks and breaches through technical security measures to ensure the ongoing confidentiality, integrity, and availability of personal data.

Among the key benefits of ATP are the following:

- Detecting the undetectable - sensors built deep into the operating system kernel, Windows security experts, and unique optics from over 1 billion machines and signals across all Microsoft services.
- Built in, not bolted on - agentless with high performance and low impact, cloud-powered; easy management with no deployment.
- Single pane of glass for Windows security - explore 6 months of rich machine timeline that unifies security events from Windows Defender ATP, Windows Defender Antivirus.
- Power of the Microsoft graph - leverages the Microsoft Intelligence Security Graph to integrate detection and exploration with Office 365 ATP subscription, to track back and respond to attacks.

Read more at [What's new in the Windows Defender ATP Creators Update preview](#).

To provide Detection capabilities, Windows 10 improves our OS memory and kernel sensors to enable detection of attackers who are employing in-memory and kernel-level attacks – shining a light into previously dark spaces where attackers hid from conventional detection tools. We've already successfully leveraged this new technology against zero-days attacks on Windows.

The screenshot shows the Windows Defender Security Center interface. The main alert is titled "Process privilege escalation due to kernel exploit" and occurred on 02/06/2017 at 18:48:52. The alert is categorized as "Privilege Escalation" and is "New". The affected process is "WINWORD.EXE".

**Alert Process Tree:**

```
graph TD
    ntoskrnl.exe --> smss.exe
    smss.exe --> winlogon.exe
    winlogon.exe --> userinit.exe
    userinit.exe --> explorer.exe
    explorer.exe --> OUTLOOK.EXE
    OUTLOOK.EXE --> WINWORD.EXE
    WINWORD.EXE --> Access token modified
    Access token modified --> WINWORD.EXE
    WINWORD.EXE --> MSASCUi.exe
    MSASCUi.exe --> lqinfo.exe
    lqinfo.exe --> OneDrive.exe
    OneDrive.exe --> LogonUI.exe
    LogonUI.exe --> forestrvhst.exe
    forestrvhst.exe --> dwm.exe
    dwm.exe --> csrss.exe
```

**Execution details for WINWORD.EXE:**

- Execution time: 02/06/2017 | 16:37:08
- Full path: C:\Program Files\Microsoft Office\Office16\WINWORD.D.EXE
- User: J\CONTOSO\Jonathan.Walcott
- Access privileges (UAC): Restricted
- Integrity level: Medium
- Process ID: 6196
- Command line: ["WINWORD.EXE" /a "C:\Users\Jonathan.Walcott\AppData\Local\Microsoft\Windows\WinSxS\xml\pt\_SortLook\138F196C1848484F\Programs - Microsoft Trainor.doc" /a]

**File details for WINWORD.EXE:**

- Sha1: [1d3626904b7127734]
- MD5: [e9313c687c3d01159]
- Sha256: [5d75d5e8bb650552]
- Size: 1.0 MB
- Signer: Microsoft Corporation
- Issuer: Microsoft Code Signing PCA

**Detections:**

- Alerts: 1 (red), 2 (yellow), 0 (green)
- Virus Total detection ratio: 0/58 not
- Windows Defender AV: No detections found

**Observed worldwide:**

- Count: 86.9k
- First seen: 2 years ago
- Last seen: 15 hours ago

We continue to upgrade our detections of ransomware and other advanced attacks, applying our behavioral and machine-learning detection library to counter changing attacks trends. Our historical detection capability ensures



new detection rules apply to up to six months of stored data to detect attacks that previously went unnoticed. Customers can also add customized detection rules or IOCs to augment the detection dictionary.

Customers asked us for a single pane of glass across the entire Windows security stack. Windows Defender Antivirus detections and Windows Defender Device Guard blocks are the first to surface in the Windows Defender ATP portal interleaved with Windows Defender ATP detections. The new user entity adds identity as a pivot, providing insight into actions, relationships, and alerts that span machines and allow us to track attackers moving laterally across the network.

Our alert page now includes a new process tree visualization that aggregates multiple detections and related events into a single view that helps security teams reduce the time to resolve cases by providing the information required to understand and resolve incidents without leaving the alert page.

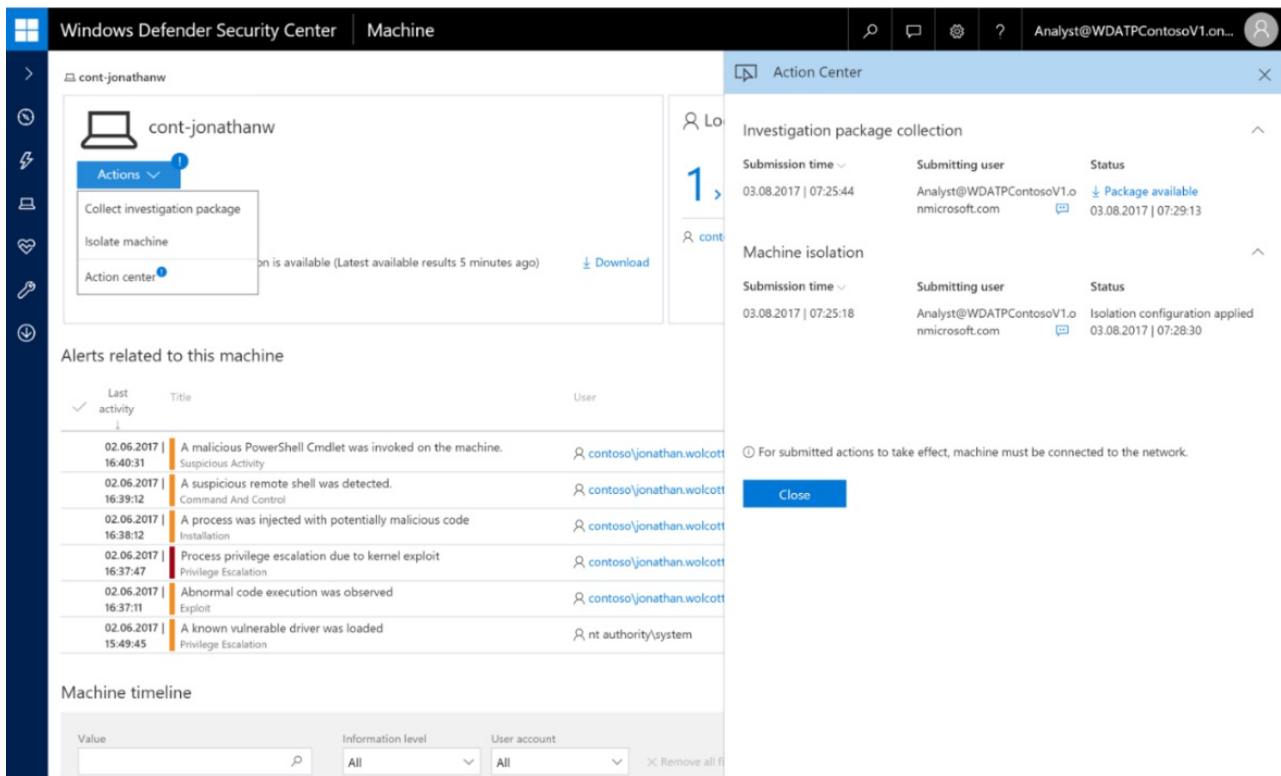
Security Operations (SecOps) can hunt for evidence of attacks, such as file names or hashes, IP addresses or URLs, behaviors, machines, or users. They can do this immediately by searching the organization's cloud inventory, across all machines – and going back up to 6 months in time – even if machines are offline, have been reimaged, or no longer exist.

The screenshot shows the Windows Defender Security Center interface for a user named Jonathan Wolcott. The user's profile includes their name, job title (Sales Manager), department (Sales), and last seen time (a month ago). A 'Logged on machines' section shows 2 machines, including 'cont-jonathanw (Local admin)'. Below this is a table of alerts related to the user, with columns for Last activity, Title, Machine, Severity, Status, and Assigned to. The alerts include suspicious sequences of exploration activities, unexpected behavior, malicious PowerShell cmdlets, suspicious PowerShell commandlines, and code executed from remote machines. At the bottom, there is a logon history chart for the organization from Feb 09 2017 to Mar 09 2017, showing logon types and a table of observed users for machines 'cont-jonathanw' and 'cont-jayhardee'.

Last activity	Title	Machine	Severity	Status	Assigned to
02.06.2017   16:41:29	Suspicious sequence of exploration activities <i>Reconnaissance</i>	cont-jayhardee	Low	New	Not assigned
02.06.2017   16:41:13	Unexpected behavior observed by a process run with no command line arguments <i>Installation</i>	cont-jayhardee	Medium	New	Not assigned
02.06.2017   16:40:48	A malicious PowerShell Cmdlet was invoked on the machine. <i>Suspicious Activity</i>	cont-jayhardee	Medium	New	Not assigned
02.06.2017   16:40:39	Suspicious Powershell commandline <i>Suspicious Activity</i>	cont-jayhardee	Medium	New	Not assigned
02.06.2017   16:40:32	Code executed from a remote machine has communicated with an abnormal IP address. <i>Lateral Movement</i>	cont-jayhardee	Medium	New	Not assigned
02.06.2017   16:40:31	A malicious PowerShell Cmdlet was invoked on the machine. <i>Suspicious Activity</i>	cont-jonathanw	Medium	New	Not assigned
02.06.2017   16:39:12	A suspicious remote shell was detected. <i>Command And Control</i>	cont-jonathanw	Medium	New	Not assigned
02.06.2017   16:38:12	A process was injected with potentially malicious code <i>Installation</i>	cont-jonathanw	Medium	New	Not assigned

Machine	Total observed users	Most frequent user	Least frequent user
cont-jonathanw	1	contoso\jonathan.wolcott (Local admin)	contoso\jonathan.wolcott (Local admin)
cont-jayhardee	2	contoso\jay.hardee (Local admin)	contoso\jonathan.wolcott (Local admin)

When detecting an attack, security teams can now take immediate action: isolate machines, ban files from the network, kill or quarantine running processes or files, or retrieve an investigation package from a machine to provide forensic evidence – with a click of a button. Because while detecting advanced attacks is important – shutting them down is even more so.



## Identity Protection

Identify and access management is another area where the GDPR has placed special emphasis by calling for mechanisms to grant and restrict access to data subject personal data (for example, role-based access, segregation of duties).

### Multi-factor protection

Biometric authentication – using your face, iris, or fingerprint to unlock your devices – is much safer than traditional passwords. You – uniquely you – plus your device are the keys to your apps, data, and even websites and services – not a random assortment of letters and numbers that are easily forgotten, hacked, or written down and pinned to a bulletin board.

Your ability to protect personal and sensitive data, that may be stored or accessed through desktop or laptops will be further enhanced by adopting advanced authentication capabilities such as Windows Hello for Business and Windows Hello companion devices. Windows Hello for Business, part of Windows 10, gives users a personal, secured experience where the device is authenticated based on their presence. Users can log in with a look or a touch, with no need for a password.

In conjunction with Windows Hello for Business, biometric authentication uses fingerprints or facial recognition and is more secure, more personal, and more convenient. If an application supports Hello, Windows 10 enables you to authenticate applications, enterprise content, and even certain online experiences without a password being stored on your device or in a network server at all. Windows Hello for Business works with the Companion Device Framework to enhance the user authentication experience. Using the Windows Hello Companion Device Framework, a companion device can provide a rich experience for Windows Hello even when biometrics are not available (for example, if the Windows 10 desktop lacks a camera for face authentication or fingerprint reader device).

There are numerous ways one can use the Windows Hello Companion Device Framework to build a great Windows unlock experience with a companion device. For example, users can:

- Work offline (for example, while traveling on a plane)
- Attach their companion device to PC via USB, touch the button on the companion device, and automatically unlock their PC.
- Carry a phone in their pocket that is already paired with their PC over Bluetooth. Upon hitting the spacebar

on their PC, their phone receives a notification. Approve it and the PC simply unlocks.

- Tap their companion device to an NFC reader to quickly unlock their PC.
- Wear a fitness band that has already authenticated the wearer. Upon approaching PC, and by performing a special gesture (like clapping), the PC unlocks.

#### Protection against attacks by isolating user credentials

As noted in the [Windows 10 Credential Theft Mitigation Guide](#), *"the tools and techniques criminals use to carry out credential theft and reuse attacks improve, malicious attackers are finding it easier to achieve their goals. Credential theft often relies on operational practices or user credential exposure, so effective mitigations require a holistic approach that addresses people, processes, and technology. In addition, these attacks rely on the attacker stealing credentials after compromising a system to expand or persist access, so organizations must contain breaches rapidly by implementing strategies that prevent attackers from moving freely and undetected in a compromised network."*

An important design consideration for Windows 10 was mitigating credential theft — in particular, derived credentials. Windows Defender Credential Guard provides significantly improved security against derived credential theft and reuse by implementing a significant architectural change in Windows designed to help eliminate hardware-based isolation attacks rather than simply trying to defend against them.

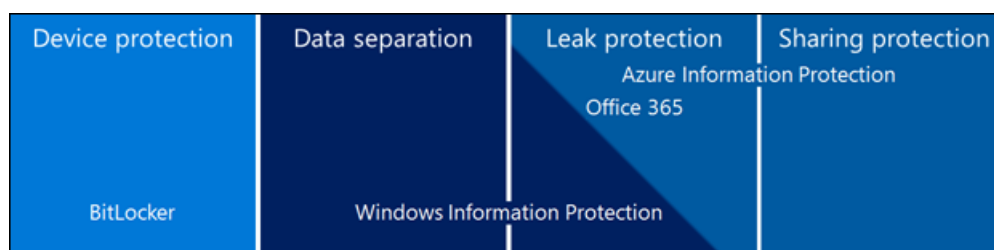
When Credential Manager domain credentials, NTLM, and Kerberos derived credentials are protected using virtualization-based security, the credential theft attack techniques and tools used in many targeted attacks are blocked. Malware running in the operating system with administrative privileges can't extract secrets that are protected by virtualization-based security. While Windows Defender Credential Guard is a powerful mitigation, persistent threat attacks will likely shift to new attack techniques and you should also incorporate Windows Defender Device Guard, as described above, and other security strategies and architectures.

#### Information Protection

The GDPR is focused on information protection regarding data that is considered as personal or sensitive in relation to a natural person, or data subject. Device protection, protection against threats, and identity protection are all important elements of a Defense in Depth strategy surrounding a layer of information protection in your laptop and desktop systems.

As to the protection of data, the GDPR recognizes that in assessing data security risk, consideration should be given to the risks that are presented such as accidental loss, unauthorized disclosure of, or access to, personal data transmitted, stored or otherwise processed. It also recommends that measures taken to maintain an appropriate level of security should consider the state-of-the-art and the costs of implementation in relation to the risks among other factors.

Windows 10 provides built in risk mitigation capabilities for today's threat landscape. In this section, we will look at the types of technologies that will help your journey toward GDPR compliance and at the same time provide you with solid overall data protection as part of a comprehensive information protection strategy.



#### Encryption for lost or stolen devices

The GDPR calls for mechanisms that implement appropriate technical security measures to confirm the ongoing confidentiality, integrity, and availability of both personal data and processing systems. BitLocker Encryption, first introduced as part of Microsoft's Next-Generation Secure Computing Base architecture in 2004 and made available with Windows Vista, is a built-in data protection feature that integrates with the operating system and

addresses the threats of data theft or exposure from lost, stolen, or inappropriately decommissioned computers.

BitLocker provides the most protection when used with a Trusted Platform Module (TPM) version 1.2 or later. The TPM is a hardware component installed in many newer computers by the computer manufacturers. It works with BitLocker to protect user data and to ensure that a computer has not been tampered with while the system was offline.

Data on a lost or stolen computer is vulnerable to unauthorized access, either by running a software-attack tool against it or by transferring the computer's hard disk to a different computer. BitLocker helps mitigate unauthorized data access by enhancing file and system protections. BitLocker also helps render data inaccessible when BitLocker-protected computers are decommissioned or recycled.

Related to BitLocker are Encrypted Hard Drives, a new class of hard drives that are self-encrypting at a hardware level and allow for full disk hardware encryption. Encrypted Hard Drives use the rapid encryption that is provided by BitLocker Drive Encryption to enhance data security and management.

By offloading the cryptographic operations to hardware, Encrypted Hard Drives increase BitLocker performance and reduce CPU usage and power consumption. Because Encrypted Hard Drives encrypt data quickly, enterprise devices can expand BitLocker deployment with minimal impact on productivity.

Some of the benefits of Encrypted Hard Drives include:

- **Better performance.** Encryption hardware, integrated into the drive controller, allows the drive to operate at full data rate with no performance degradation.
- **Strong security based in hardware.** Encryption is always "on" and the keys for encryption never leave the hard drive. User authentication is performed by the drive before it will unlock, independently of the operating system
- **Ease of use.** Encryption is transparent to the user because it is on by default. There is no user interaction needed to enable encryption. Encrypted Hard Drives are easily erased using on-board encryption key; there is no need to re-encrypt data on the drive.
- **Lower cost of ownership.** There is no need for new infrastructure to manage encryption keys, since BitLocker leverages your Active Directory Domain Services infrastructure to store recovery information. Your device operates more efficiently because processor cycles don't need to be used for the encryption process.

#### **Preventing accidental data leaks to unauthorized users**

Part of the reality of your operating in a mobile-first, cloud-first world is the notion that some laptops will have multiple purposes – both business and personal. Yet that data that is considered as personal and sensitive regarding EU residents considered as "data subjects" must be protected in line with the requirements of the GDPR.

Windows Information Protection helps people separate their work and personal data and keeps data encrypted wherever it's stored. Your employees can safely use both work and personal data on the same device without switching applications. Windows Information Protection helps end users avoid inadvertent data leaks by sending a warning when copy/pasting information in non-corporate applications – end users can still proceed but the action will be logged centrally.

For example, employees can't send protected work files from a personal email account instead of their work account. They also can't accidentally post personal or sensitive data from a corporate site into a tweet. Windows Information Protection also helps ensure that they aren't saving personal or sensitive data in a public cloud storage location.

#### **Capabilities to classify, assign permissions and share data**

Windows Information Protection is designed to coexist with advanced data loss prevention (DLP) capabilities found in Office 365 ProPlus, Azure Information Protection, and Azure Rights Management. Advanced DLP prevents printing, for example, or protects work data that is emailed outside your company.

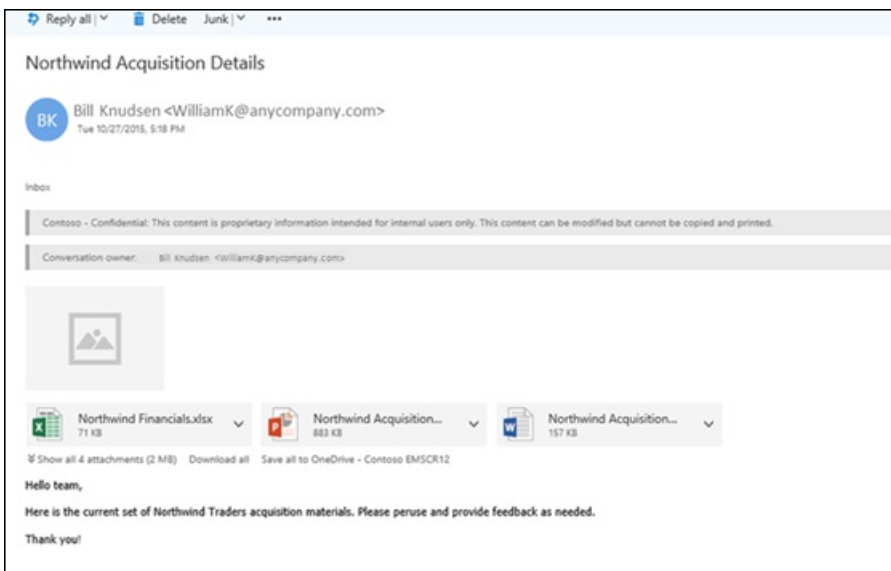
To continuously protect your data, regardless of where it is stored, with whom it is shared, or if the device is running iOS, Android or Windows, the classification and protection needs to be built into the file itself, so this protection can travel with the data wherever it goes. Microsoft Azure Information Protection (AIP) is designed to provide this persistent data protection both on-premises and in the cloud.

Data classification is an important part of any data governance plan. Adopting a classification scheme that applies throughout your business can be particularly helpful in responding to what the GDPR calls data subject (for example, your EU employee or customer) requests, because it enables enterprises to identify more readily and process personal data requests.

Azure Information Protection can be used to help you classify and label your data at the time of creation or modification. Protection in the form of encryption, which the GDPR recognizes may be appropriate at times, or visual markings can then be applied to data needing protection.

With Azure Information Protection, you can either query for data marked with a sensitivity label or intelligently identify sensitive data when a file or email is created or modified. Once identified, you can automatically classify and label the data – all based on the company's desired policy.

Azure Information Protection also helps your users share sensitive data in a secure manner. In the example below, information about a sensitive acquisition was encrypted and restricted to a group of people who were granted only a limited set of permissions on the information – they could modify the content but could not copy or print it.



## Related content for associated Windows 10 solutions

- **Windows Hello for Business:** <https://www.youtube.com/watch?v=WOvoXQdj-9E> and <https://docs.microsoft.com/en-us/windows/access-protection/hello-for-business/hello-identity-verification>
- **Windows Defender Antivirus:** <https://www.youtube.com/watch?v=P1aNEy09NaI> and <https://docs.microsoft.com/en-us/windows/threat-protection/windows-defender-antivirus/windows-defender-antivirus-in-windows-10>
- **Windows Defender Advanced Threat Protection:** <https://www.youtube.com/watch?v=qxeGa3pxlwg> and <https://docs.microsoft.com/en-us/windows/threat-protection/windows-defender-atp/windows-defender-advanced-threat-protection>
- **Windows Defender Device Guard:** <https://www.youtube.com/watch?v=F-pTkesjkhI> and <https://docs.microsoft.com/en-us/windows/device-security/device-guard/device-guard-deployment-guide>
- **Windows Defender Credential Guard:** <https://www.youtube.com/watch?v=F-pTkesjkhI> and <https://docs.microsoft.com/en-us/windows/access-protection/credential-guard/credential-guard>

- **Windows Information Protection:** <https://www.youtube.com/watch?v=wLkQOmK7-Jg> and <https://docs.microsoft.com/en-us/windows/threat-protection/windows-information-protection/protect-enterprise-data-using-wip>
- Windows 10 Security Guide: <https://technet.microsoft.com/en-us/itpro/windows/keep-secure/windows-10-security-guide>

## Disclaimer

This article is a commentary on the GDPR, as Microsoft interprets it, as of the date of publication. We've spent a lot of time with GDPR and like to think we've been thoughtful about its intent and meaning. But the application of GDPR is highly fact-specific, and not all aspects and interpretations of GDPR are well-settled.

As a result, this article is provided for informational purposes only and should not be relied upon as legal advice or to determine how GDPR might apply to you and your organization. We encourage you to work with a legally-qualified professional to discuss GDPR, how it applies specifically to your organization, and how best to ensure compliance.

MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED, OR STATUTORY, AS TO THE INFORMATION IN THIS ARTICLE. This article is provided "as-is." Information and views expressed in this article, including URL and other Internet website references, may change without notice.

This article does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this article for your internal, reference purposes only.

Published September 2017

Version 1.0

© 2017 Microsoft. All rights reserved.

# Windows 10 and the GDPR for IT Decision Makers

7/2/2018 • 14 minutes to read • [Edit Online](#)

Applies to:

- Windows 10, version 1803
- Windows 10, version 1709
- Windows 10, version 1703

This topic provides IT Decision Makers with a basic understanding of the relationship between users in an organization and Microsoft in the context of the GDPR (General Data Protection Regulation). You will also learn what role an IT organization plays for that relationship.

For more information about the GDPR, see:

- [Microsoft GDPR Overview](#)
- [Microsoft Trust Center FAQs about the GDPR](#)
- [Microsoft Service Trust Portal \(STP\)](#)
- [Get Started: Support for GDPR Accountability](#)

## GDPR fundamentals

Here are some GDPR fundamentals:

- On May 25, 2018, this EU data privacy law is implemented. It sets a new global bar for data privacy rights, security, and compliance.
- The GDPR is fundamentally about protecting and enabling the privacy rights of individuals – both customers and employees.
- The European law establishes strict global data privacy requirements governing how organizations manage and protect personal data while respecting individual choice – no matter where data is sent, processed, or stored.
- A request by an individual to an organization to take an action on their personal data is referred to here as a *data subject request*, or *DSR*.

Microsoft believes data privacy is a fundamental right, and that the GDPR is an important step forward for clarifying and enabling individual privacy rights. We also recognize that the GDPR requires significant changes by organizations all over the world with regard to the discovery, management, protection, and reporting of personal data that is collected, processed, and stored within an organization.

### What is personal data under the GDPR?

Article 4 (1) of [the GDPR](#) defines personal data as any information relating to an identified or identifiable person. There is no distinction between a person's private, public, or work roles. As defined by the GDPR, personal data includes, but is not limited to:

- Name
- Email address
- Credit card numbers
- IP addresses
- Social media posts
- Location information
- Handwriting patterns

- Voice input to cloud-based speech services

## Controller and processor under the GDPR: Who does what

### Definition

The GDPR describes specific requirements for allocating responsibility for controller and processor activities related to personal data. Thus, every organization that processes personal data must determine whether it is acting as a controller or processor for a specific scenario.

- **Controller:** GDPR Article 4 (7) defines the 'controller' as the natural or legal person, public authority, agency or other body which, alone or jointly with others, determines the purposes and means of the processing of personal data.
- **Processor:** According to the GDPR Article 4 (8) 'processor' means a natural or legal person, public authority, agency or other body which processes personal data on behalf of the controller.

### Controller scenario

For example, when an organization is using Microsoft Windows Defender Advanced Threat Protection (ATP) to detect, investigate, and respond to advanced threats on their networks as part of their IT operations, that organization is collecting data from the user's device – data, that might include personal data. In this scenario, the organization is the *controller* of the respective personal data, since the organization controls the purpose and means of the processing for data being collected from the devices that have Windows Defender ATP enabled.

### Processor scenario

In the controller scenario described above, Microsoft is a *processor* because Microsoft provides data processing services to that controller (in the given example, an organization that subscribed to Windows Defender ATP and enabled it for the user's device). As processor, Microsoft only processes data on behalf of the enterprise customer and does not have the right to process data beyond their instructions as specified in a written contract, such as the [Microsoft Product Terms and the Microsoft Online Services Terms \(OST\)](#).

## GDPR relationship between a Windows 10 user and Microsoft

For Windows 10 services, Microsoft usually is the controller (with exceptions, such as Windows Defender ATP). The following sections describe what that means for the related data.

### Types of data exchanged with Microsoft

Microsoft collects data from or generates data through interactions with users of Windows 10 devices. This information can contain personal data, as defined in [Article 4 \(1\) of the GDPR](#), that may be used to provide, support, and improve Windows 10 services.

Microsoft discloses data collection and privacy practices in detail, for example:

- As part of the Windows 10 installation;
- In the Windows 10 privacy settings;
- Via the web-based [Microsoft Privacy dashboard](#); and
- In the [Microsoft Privacy Statement](#).

It is important to differentiate between two distinct types of data Windows services are dealing with.

### Windows functional data

A user action, such as performing a Skype call, usually triggers the collection and transmission of Windows *functional data*. Some Windows components and applications connecting to Microsoft services also exchange Windows functional data to provide user functionality.

Some other examples of Windows functional data:

- The Weather app which uses the device's location to retrieve local weather or community news.
- Wallpaper and desktop settings that are synchronized across multiple devices.



For more info on how IT Professionals can manage Windows functional data sent from an organization to Microsoft, see [Manage connections from Windows operating system components to Microsoft services](#).

#### **Windows diagnostic data**

Windows diagnostic data is used to keep the operating system secure and up-to-date, troubleshoot problems, and make product improvements. The data is encrypted before being sent back to Microsoft.

Some examples of diagnostic data include:

- The type of hardware being used, information about installed apps and usage details, and reliability data on drivers running on the device.
- For users who have turned on “Tailored experiences”, it can be used to offer personalized tips, ads, and recommendations to enhance Microsoft products and services for the needs of the user.

To find more about what information is collected, how it is handled, and the available Windows diagnostic data levels, see [Understanding Windows diagnostic data](#) and [Configure Windows diagnostic data in your organization](#).

#### **IMPORTANT**

Other Microsoft services as well as 3rd party applications and drivers running on Windows devices may implement their own functionality, independently from Windows, to transport their diagnostic data to the respective publisher. Please contact them for further guidance on how to control the diagnostic data collection level and transmission of these publishers.

#### **Windows services where Microsoft is the processor under the GDPR**

Most Windows 10 services are controller services in terms of the GDPR – for both Windows functional data and Windows diagnostic data. But there are a few Windows services where Microsoft is a processor for functional data under the GDPR, such as [Windows Analytics](#) and [Windows Defender Advanced Threat Protection \(ATP\)](#).

#### **NOTE**

Both Windows Analytics and Windows Defender ATP are subscription services for organizations. Some functionality requires a certain license (please see [Compare Windows 10 editions](#)).

#### **Windows Analytics**

[Windows Analytics](#) is a service that provides rich, actionable information for helping organizations to gain deep insights into the operational efficiency and health of the Windows devices in their environment. It uses Windows diagnostic data from devices enrolled by the IT organization of an enterprise into the Windows Analytics service.

Windows [transmits Windows diagnostic data](#) to Microsoft datacenters, where that data is analyzed and stored. With Windows Analytics, the IT organization can then view the analyzed data to detect and fix issues or to improve their processes for upgrading to Windows 10.

As a result, in terms of the GDPR, the organization that has subscribed to Windows Analytics is acting as the controller, while Microsoft is the processor for Windows Analytics.

#### **NOTE**

The IT organization must explicitly enable Windows Analytics for a device after the organization subscribes.

### IMPORTANT

Windows Analytics does not collect Windows Diagnostic data by itself. Instead, Windows Analytics only uses a subset of Windows Diagnostic data that is collected by Windows for a particular device. The Windows Diagnostic data collection is controlled by the IT department of an organization or the user of a device.

### Windows Defender ATP

[Windows Defender ATP](#) is cloud-based service that collects and analyzes usage data from an organization's devices to detect security threats. Some of the data can contain personal data as defined by the GDPR. Enrolled devices transmit usage data to Microsoft datacenters, where that data is analyzed, processed, and stored. The security operations center (SOC) of the organization can view the analyzed data using the [Windows Defender ATP portal](#).

As a result, in terms of the GDPR, the organization that has subscribed to Windows Defender ATP is acting as the controller, while Microsoft is the processor for Windows Defender ATP.

### NOTE

The IT organization must explicitly enable Windows Defender ATP for a device after the organization subscribes.

### At a glance – Windows 10 services GDPR mode of operations

The following table lists in what GDPR mode – controller or processor – Windows 10 services are operating.

SERVICE	MICROSOFT GDPR MODE OF OPERATION
Windows Functional data	Controller
Windows Diagnostic data	Controller
Windows Analytics	Processor
Windows Defender Advanced Threat Detection (ATP)	Processor

Table 1: Windows 10 GDPR modes of operations for different Windows 10 services

## Recommended diagnostic data level settings

Windows diagnostic data collection level can be set by a user in Windows (*Start > Settings > Privacy > Diagnostics & feedback*) or by the IT department of an organization, using Group Policy or Mobile Device Management (MDM) techniques.

- For Windows 10, version 1803, Microsoft recommends setting the Windows diagnostic level to "Enhanced". This enables organizations to get the full functionality of [Windows Analytics](#). Those organizations who wish to share the smallest set of events for Windows Analytics can use the "Limit Enhanced diagnostic data to the minimum required by Windows Analytics" filtering mechanism that Microsoft introduced in Windows 10, version 1709. When enabled, this feature limits the operating system diagnostic data events included in the Enhanced level to the smallest set of data required by Windows Analytics.

### NOTE

For more information on the Enhanced level, see [Configure Windows diagnostic data in your organization](#).

- For Windows 10, version 1709, and Windows 10, version 1703, the recommended Windows diagnostic

level configuration for EEA and Switzerland commercial users is “Basic”.

- For Windows 7, Microsoft recommends configuring enterprise devices for Windows Analytics to facilitate upgrade planning to Windows 10.

## Controlling the data collection and notification about it

Windows 10 sends diagnostic data to Microsoft services, and some of that data can contain personal data. Both the user and the IT organization have the ability to control the transmission of that data to Microsoft.

### Adjusting privacy settings by the user

A user has the ability to adjust additional privacy settings in Windows by navigating to *Start > Settings > Privacy*. For example, a user can control if location is enabled or disabled, whether or not to transmit feedback on inking and typing input to Microsoft for improving the personal accuracy of these services, or if Windows collects activities for syncing it with other devices.

For a standard user in an organization, some privacy settings might be controlled by their IT department. This is done using Group Policies or Mobile Device Management (MDM) settings. If this is the case, the user will see an alert that says ‘Some settings are hidden or managed by your organization’ when they navigate to *Start > Settings > Privacy*. As such, the user can only change some settings, but not all.

### Users can lower the diagnostic level

Starting with Windows 10, version 1803, a user can change the Windows diagnostics data level for their device below to what was set by their IT department. Organizations can allow or disallow this feature by configuring the Group Policy **Computer Configuration\Administrative Templates\Windows Components\Data Collection and Preview Builds\Configure telemetry opt-in setting user interface** or the MDM policy **ConfigureTelemetryOptInSettingsUx**.

If an IT organization has not disabled this policy, users within the organization can change their own Windows diagnostic data collection level in *Start > Settings > Privacy > Diagnostics & feedback*. For example, if the IT organization enabled this policy and set the level to “Full”, a user can modify the Windows diagnostics data level setting to “Basic”.

### Notification at logon

Windows 10, version 1803, and later can provide users with a notification during their logon. If the IT organization has not disabled the Group Policy **Computer Configuration\Administrative Templates\Windows Components\Data Collection and Preview Builds\Configure telemetry opt-in change notifications** or the MDM policy **ConfigureTelemetryOptInChangeNotification**, Windows diagnostic data notifications can appear at logon so that the users of a device are aware of the data collection.

This notification can also be shown when the diagnostic level for the device was changed. For instance, if the diagnostic level on the device is set to “Basic” and the IT organization changes it to “Full”, users will be notified on their next logon.

### Diagnostic Data Viewer (DDV)

In Windows 10, version 1803 and later, users can invoke the [Diagnostic Data Viewer \(DDV\)](#) to see what Windows diagnostic data is collected on their local device. This app lets a user review the diagnostic data collected on his device that is being sent to Microsoft. The DDV groups the information into simple categories based on how it is used by Microsoft.

A user can turn on Windows diagnostic data viewing by going to go to *Start > Settings > Privacy > Diagnostics & feedback*. Under the ‘Diagnostic data viewer’ section, the user has to enable the ‘If data viewing is enabled, you can see your diagnostics data’ option. After DDV is installed on the device, the user can start it by clicking the ‘Diagnostic Data Viewer’ in the ‘Diagnostic data viewer’ section of *Start > Settings > Privacy > Diagnostics & feedback*.

Also, the user can delete all Windows diagnostic data collected from the device. This is done by clicking the 'Delete' button in the 'Delete diagnostic data' section of *Start > Settings > Privacy > Diagnostics & feedback*.

### **Windows 10 personal data services configuration**

Microsoft assembled a list of Windows 10 services configuration settings that are useful for personal data privacy protection and related regulations, such as the General Data Protection Regulation (GDPR). There is one section with settings for service data that is managed at Microsoft and a section for local data that is managed by an IT organization.

IT Professionals that are interested in this configuration, see [Windows 10 personal data services configuration](#).

### **Windows 10 connections to Microsoft**

To find out more about the network connections that Windows components make to Microsoft as well as the privacy settings that affect data shared with either Microsoft or apps, see [Manage connections from Windows operating system components to Microsoft services](#) and [Manage Windows 10 connection endpoints](#). These articles describe how these settings can be managed by an IT Professional.

## **At-a-glance: the relationship between an IT organization and the GDPR**

Because Microsoft is a controller for data collected by Windows 10, the user can work with Microsoft to satisfy GDPR requirements. While this relationship between Microsoft and a user is evident in a consumer scenario, an IT organization can influence that relationship in an enterprise scenario. For example, the IT organization has the ability to centrally configure the Windows diagnostic data level by using Group Policy or MDM settings.

## **Further reading**

### **Optional settings / features that further improve the protection of personal data**

Personal data protection is one of the goals of the GDPR. One way of improving personal data protection is to use the modern and advanced security features of Windows 10. An IT organization can learn more at [Mitigate threats by using Windows 10 security features](#) and [Standards for a highly secure Windows 10 device](#).

#### **NOTE**

Some of these features might require a particular Windows hardware, such as a computer with a Trusted Platform Module (TPM) chip, and can depend on a particular Windows product (such as Windows 10 E5).

### **Windows Security Baselines**

Microsoft has created Windows Security Baselines to efficiently configure Windows 10. For more information, please visit [Windows Security Baselines](#).

### **Windows Restricted Traffic Limited Functionality Baseline**

To make it easier to deploy settings that restrict connections from Windows 10 to Microsoft, IT Professionals can apply the Windows Restricted Traffic Limited Functionality Baseline, available [here](#).

#### **IMPORTANT**

Some of the settings of the Windows Restricted Traffic Limited Functionality Baseline will reduce the functionality and security configuration of a device in the organization and are therefore not recommended.

### **Microsoft Trust Center and Service Trust Portal**

Please visit our [GDPR section of the Microsoft Trust Center](#) to obtain additional resources and to learn more about how Microsoft can help you fulfill specific GDPR requirements. There you can find lots of useful information about the GDPR, including how Microsoft is helping customers to successfully master the GDPR, a FAQ list, and a list of

[resources for GDPR compliance](#). Also, please check out the [Compliance Manager](#) of the Microsoft [Service Trust Portal \(STP\)](#) and [Get Started: Support for GDPR Accountability](#).

### **Additional resources**

#### **FAQs**

- [Windows 10 feedback, diagnostics, and privacy](#)
- [Microsoft Edge and privacy](#)
- [Windows Hello and privacy](#)
- [Wi-Fi Sense](#)

#### **Blogs**

- [Privacy and Windows 10](#)

#### **Privacy Statement**

- [Microsoft Privacy Statement](#)

#### **Other resources**

- [Privacy at Microsoft](#)

# Windows 10 personal data services configuration

7/13/2018 • 5 minutes to read • [Edit Online](#)

Applies to:

- Windows 10, version 1803

Microsoft assembled a list of Windows 10 services configuration settings that are useful for personal data privacy protection and related regulations, such as the General Data Protection Regulation (GDPR). There is one section with settings for service data that is managed at Microsoft and a section for local data that is managed by an IT organization.

IT Professionals that are interested in applying these settings via group policies can find the configuration for download [here](#).

## Introduction

Microsoft collects data from or generates it through interactions with users of Windows 10 devices. This information can contain personal data that may be used to provide, support, and improve Windows 10 services.

Many Windows 10 services are controller services. A user can manage data collection settings, for example by opening *Start > Settings > Privacy* or by visiting the [Microsoft Privacy dashboard](#). While this relationship between Microsoft and a user is evident in a consumer type scenario, an IT organization can influence that relationship. For example, the IT department has the ability to configure the Windows diagnostic data level across their organization by using Group Policy, registry, or Mobile Device Management (MDM) settings.

Below is a collection of settings related to the Windows 10 personal data services configuration that IT Professionals can use as guidance for influencing Windows diagnostic data collection and personal data protection.

## Windows diagnostic data

Windows 10 collects Windows diagnostic data—such as usage data, performance data, inking, typing, and utterance data—and sends it back to Microsoft. That data is used for keeping the operating system secure and up-to-date, to troubleshoot problems, and to make product improvements. For users who have turned on "Tailored experiences", that data can also be used to offer personalized tips, ads, and recommendations to enhance Microsoft products and services for your needs.

The following options for configuring Windows diagnostic data are relevant in this context.

### Diagnostic level

This setting determines the amount of Windows diagnostic data sent to Microsoft.

#### NOTE

In Windows 10, version 1709, Microsoft introduced a new feature: "Limit Enhanced diagnostic data to the minimum required by Windows Analytics". When enabled, this feature limits the operating system diagnostic data events included in the Enhanced level to the smallest set of data required by [Windows Analytics](#). For more information on the Enhanced level, see [Configure Windows diagnostic data in your organization](#).

<b>Group Policy</b>	Computer Configuration\Administrative Templates\Windows Components\Data Collection and Preview Builds
<b>Policy Name</b>	Allow Telemetry
<b>Default setting</b>	2 - Enhanced
<b>Recommended</b>	2 - Enhanced

<b>Group Policy</b>	User Configuration\Administrative Templates\Windows Components\Data Collection and Preview Builds
<b>Policy Name</b>	Allow Telemetry
<b>Default setting</b>	2 - Enhanced
<b>Recommended</b>	2 - Enhanced

#### Registry

<b>Registry key</b>	HKLM\Software\Policies\Microsoft\Windows\DataCollection
<b>Value</b>	AllowTelemetry
<b>Type</b>	REG_DWORD
<b>Setting</b>	"00000002"

<b>Registry key</b>	HKCU\Software\Policies\Microsoft\Windows\DataCollection
<b>Value</b>	AllowTelemetry
<b>Type</b>	REG_DWORD
<b>Setting</b>	"00000002"

#### MDM

<b>MDM CSP</b>	System
<b>Policy</b>	AllowTelemetry (scope: device and user)
<b>Default setting</b>	2 – Enhanced
<b>Recommended</b>	2 – Allowed

#### Diagnostic opt-in change notifications

This setting determines whether a device shows notifications about Windows diagnostic data levels to people on first logon or when changes occur in the diagnostic configuration.

#### Group Policy

<b>Group Policy</b>	Computer Configuration\Administrative Templates\Windows Components\Data Collection and Preview Builds
<b>Policy Name</b>	Configure telemetry opt-in change notifications
<b>Default setting</b>	Enabled
<b>Recommended</b>	Enabled

#### Registry

<b>Registry key</b>	HKLM\Software\Policies\Microsoft\Windows\DataCollection
<b>Value</b>	DisableTelemetryOptInChangeNotification
<b>Type</b>	REG_DWORD
<b>Setting</b>	"00000000"

#### MDM

<b>MDM CSP</b>	System
<b>Policy</b>	ConfigureTelemetryOptInChangeNotification
<b>Default setting</b>	0 – Enabled
<b>Recommended</b>	0 – Enabled

#### Configure telemetry opt-in setting user interface

This setting determines whether people can change their own Windows diagnostic data level in in *Start > Settings > Privacy > Diagnostics & feedback*.

#### Group Policy

<b>Group Policy</b>	Computer Configuration\Administrative Templates\Windows Components\Data Collection and Preview Builds
<b>Policy Name</b>	Configure telemetry opt-in setting user interface
<b>Default setting</b>	Enabled
<b>Recommended</b>	Enabled

#### Registry



<b>Registry key</b>	HKLM\Software\Policies\Microsoft\Windows\DataCollection
<b>Value</b>	DisableTelemetryOptInSettingsUx
<b>Type</b>	REG_DWORD
<b>Setting</b>	"00000001"

#### MDM

<b>MDM CSP</b>	System
<b>Policy</b>	ConfigureTelemetryOptInSettingsUx
<b>Default setting</b>	0 – Enabled
<b>Recommended</b>	0 – Enabled

## Policies affecting personal data protection managed by the Enterprise IT

There are additional settings usually managed by the Enterprise IT that also affect the protection of personal data.

The following options for configuring these policies are relevant in this context.

### BitLocker

The following settings determine whether fixed and removable drives are protected by the BitLocker Drive Encryption.

#### Fixed Data Drives

##### Group Policy

<b>Group Policy</b>	Computer Configuration\Administrative Templates\Windows Components\Bitlocker Drive Encryption\Fixed Data Drives
<b>Policy Name</b>	Deny write access to fixed drives not protected by BitLocker
<b>Default setting</b>	Not configured
<b>Recommended</b>	Enabled

#### Registry

<b>Registry key</b>	HKLM\System\CurrentControlSet\Policies\Microsoft\FVE
<b>Value</b>	FDVDenyWriteAccess
<b>Type</b>	REG_DWORD

<b>Setting</b>	"00000001"

#### MDM

<b>MDM CSP</b>	BitLocker
<b>Policy</b>	RemovableDrivesRequireEncryption
<b>Default setting</b>	Disabled
<b>Recommended</b>	Enabled (see <a href="#">instructions</a> )

#### Removable Data Drives

##### Group Policy

<b>Group Policy</b>	Computer Configuration\Administrative Templates\Windows Components\Bitlocker Drive Encryption\Removable Data Drives
<b>Policy Name</b>	Deny write access to removable drives not protected by BitLocker
<b>Default setting</b>	Not configured
<b>Recommended</b>	Enabled

#### Registry

<b>Registry key</b>	HKLM\System\CurrentControlSet\Policies\Microsoft\FVE
<b>Value</b>	RDVDenyWriteAccess
<b>Type</b>	REG_DWORD
<b>Setting</b>	"00000001"

<b>Registry key</b>	HKLM\Software\Policies\Microsoft\FVE
<b>Value</b>	RDVDenyCrossOrg
<b>Type</b>	REG_DWORD
<b>Setting</b>	"00000000"

#### MDM

<b>MDM CSP</b>	BitLocker

<b>Policy</b>	RemovableDrivesRequireEncryption
<b>Default setting</b>	Disabled
<b>Recommended</b>	Enabled (see <a href="#">instructions</a> )

### Privacy – AdvertisingID

This setting determines if the advertising ID, which preventing apps from using the ID for experiences across apps, is turned off.

#### Group Policy

<b>Group Policy</b>	Computer Configuration\Administrative Templates\System\User Profiles
<b>Policy Name</b>	Turn off the advertising ID
<b>Default setting</b>	Not configured
<b>Recommended</b>	Enabled

#### Registry

<b>Registry key</b>	HKLM\Software\Policies\Microsoft\Windows\AdvertisingInfo
<b>Value</b>	DisabledByGroupPolicy
<b>Type</b>	REG_DWORD
<b>Setting</b>	"00000001"

#### MDM

<b>MDM CSP</b>	Privacy
<b>Policy</b>	DisableAdvertisingId
<b>Default setting</b>	65535 (default) - Not configured
<b>Recommended</b>	1 – Enabled

### Edge

These settings whether employees send "Do Not Track" from the Microsoft Edge web browser to websites.

#### NOTE

Please see [this Microsoft blog post](#) for more details on why the "Do Not Track" is no longer the default setting.

#### Group Policy

<b>Group Policy</b>	Computer Configuration\Administrative Templates\Windows Components\Microsoft Edge
<b>Policy Name</b>	Configure Do Not Track
<b>Default setting</b>	Disabled
<b>Recommended</b>	Disabled

<b>Group Policy</b>	User Configuration\Administrative Templates\Windows Components\Microsoft Edge
<b>Policy Name</b>	Configure Do Not Track
<b>Default setting</b>	Disabled
<b>Recommended</b>	Disabled

#### Registry

<b>Registry key</b>	HKLM\Software\Policies\Microsoft\MicrosoftEdge\Main
<b>Value</b>	DoNotTrack
<b>Type</b>	REG_DWORD
<b>Setting</b>	"00000000"

<b>Registry key</b>	HKCU\Software\Policies\Microsoft\MicrosoftEdge\Main
<b>Value</b>	DoNotTrack
<b>Type</b>	REG_DWORD
<b>Setting</b>	"00000000"

#### MDM

<b>MDM CSP</b>	Browser
<b>Policy</b>	AllowDoNotTrack (scope: device + user)
<b>Default setting</b>	0 (default) – Not allowed
<b>Recommended</b>	0 – Not allowed

#### Internet Explorer

These settings whether employees send "Do Not Track" header from the Microsoft Explorer web browser to websites.

**Group Policy**

<b>Group Policy</b>	Computer Configuration\Administrative Templates\Windows Components\Internet Explorer\Internet Control Panel\Advanced Page
<b>Policy Name</b>	Always send Do Not Track header
<b>Default setting</b>	Disabled
<b>Recommended</b>	Disabled

<b>Group Policy</b>	User Configuration\Administrative Templates\Windows Components\Internet Explorer\Internet Control Panel\Advanced Page
<b>Policy Name</b>	Always send Do Not Track header
<b>Default setting</b>	Disabled
<b>Recommended</b>	Disabled

**Registry**

<b>Registry key</b>	HKLM\Software\Policies\Microsoft\Internet Explorer\Main
<b>Value</b>	DoNotTrack
<b>Type</b>	REG_DWORD
<b>Setting</b>	"00000000"

<b>Registry key</b>	HKCU\Software\Policies\Microsoft\Internet Explorer\Main
<b>Value</b>	DoNotTrack
<b>Type</b>	REG_DWORD
<b>Setting</b>	"00000000"

**MDM**

<b>MDM CSP</b>	N/A
----------------	-----

# Additional resources

## FAQs

- [Windows 10 feedback, diagnostics, and privacy](#)
- [Microsoft Edge and privacy](#)
- [Windows Hello and privacy](#)
- [Wi-Fi Sense](#)

## Blogs

- [Privacy and Windows 10](#)

## Privacy Statement

- [Microsoft Privacy Statement](#)

## Windows Privacy on docs.microsoft.com

- [Manage connections from Windows operating system components to Microsoft services](#)
- [Manage Windows 10 connection endpoints](#)
- [Understanding Windows diagnostic data](#)
- [Configure Windows diagnostic data in your organization](#)

## Other resources

- [Privacy at Microsoft](#)

# Configure Windows diagnostic data in your organization

6/29/2018 • 28 minutes to read • [Edit Online](#)

## Applies to

- Windows 10 Enterprise
- Windows 10 Mobile
- Windows Server

At Microsoft, we use Windows diagnostic data to inform our decisions and focus our efforts in providing the most robust, most valuable platform for your business and the people who count on Windows to enable them to be as productive as possible. Diagnostic data gives users a voice in the operating system's development. This guide describes the importance of Windows diagnostic data and how we protect that data. Additionally, it differentiates between diagnostic data and functional data. It also describes the diagnostic data levels that Windows supports. Of course, you can choose how much diagnostic data is shared with Microsoft, and this guide demonstrates how.

To frame a discussion about diagnostic data, it is important to understand Microsoft's privacy principles. We earn customer trust every day by focusing on six key privacy principles as described at [privacy.microsoft.com](https://privacy.microsoft.com). These principles guided the implementation of the Windows diagnostic data system in the following ways:

- **Control.** We offer customers control of the diagnostic data they share with us by providing easy-to-use management tools.
- **Transparency.** We provide information about the diagnostic data that Windows and Windows Server collects so our customers can make informed decisions.
- **Security.** We encrypt diagnostic data in transit from your device via TLS 1.2, and additionally use certificate pinning to secure the connection.
- **Strong legal protections.** We respect customers' local privacy laws and fight for legal protection of their privacy as a fundamental human right.
- **No content-based targeting.** We take steps to avoid and minimize the collection of customer content, such as the content of files, chats, or emails, through the Windows diagnostic data system. Customer content inadvertently collected is kept confidential and not used for user targeting.
- **Benefits to you.** We collect Windows diagnostic data to help provide you with an up-to-date, more secure, reliable and performant product, and to improve Windows for all our customers.

This article applies to Windows and Windows Server diagnostic data only. Other Microsoft or third-party apps, such as System Center Configuration Manager, System Center Endpoint Protection, or System Center Data Protection Manager, might send data to their cloud services in ways that are inconsistent with this guide. Their publishers are responsible for notifying users of their privacy policies, diagnostic data controls, and so on. This article describes the types of diagnostic data we may gather, the ways you might manage it in your organization, and some examples of how diagnostic data can provide you with valuable insights into your enterprise deployments. Microsoft uses the data to quickly identify and address issues affecting its customers.

Use this article to make informed decisions about how you might configure diagnostic data in your organization. Diagnostic data is a term that means different things to different people and organizations. For this article, we discuss diagnostic data as system data that is uploaded by the Connected User Experiences and Telemetry component. The diagnostic data is used to help keep Windows devices secure by identifying malware trends and other threats and to help Microsoft improve the quality of Windows and Microsoft services.

We are always striving to improve our documentation and welcome your feedback. You can provide feedback by contacting [telmhelp@microsoft.com](mailto:telmhelp@microsoft.com).

## Overview

In previous versions of Windows and Windows Server, Microsoft used diagnostic data to check for updated or new Windows Defender signatures, check whether Windows Update installations were successful, gather reliability information through the Reliability Analysis Component (RAC), and gather reliability information through the Windows Customer Experience Improvement Program (CEIP) on Windows. In Windows 10 and Windows Server 2016, you can control diagnostic data streams by using the Privacy option in Settings, Group Policy, or MDM.

For Windows 10, we invite IT pros to join the [Windows Insider Program](#) to give us feedback on what we can do to make Windows work better for your organization.

## Understanding Windows diagnostic data

Windows as a Service is a fundamental change in how Microsoft plans, builds, and delivers the operating system. Historically, we released a major Windows version every few years. The effort required to deploy large and infrequent Windows versions was substantial. That effort included updating the infrastructure to support the upgrade. Windows as a Service accelerates the cadence to provide rich updates more frequently, and these updates require substantially less effort to roll out than earlier versions of Windows. Since it provides more value to organizations in a shorter timeframe, delivering Windows as a Service is a top priority for us.

The release cadence of Windows may be fast, so feedback is critical to its success. We rely on diagnostic data at each stage of the process to inform our decisions and prioritize our efforts.

### What is Windows diagnostic data?

Windows diagnostic data is vital technical data from Windows devices about the device and how Windows and related software are performing. It's used in the following ways:

- Keep Windows up to date
- Keep Windows secure, reliable, and performant
- Improve Windows – through the aggregate analysis of the use of Windows
- Personalize Windows engagement surfaces

Here are some specific examples of Windows diagnostic data:

- Type of hardware being used
- Applications installed and usage details
- Reliability information on device drivers

### What is NOT diagnostic data?

Diagnostic data can sometimes be confused with functional data. Some Windows components and apps connect to Microsoft services directly, but the data they exchange is not diagnostic data. For example, exchanging a user's location for local weather or news is not an example of diagnostic data—it is functional data that the app or service requires to satisfy the user's request.

There are subtle differences between diagnostic data and functional data. Windows collects and sends diagnostic data in the background automatically. You can control how much information is gathered by setting the diagnostic data level. Microsoft tries to avoid collecting personal information wherever possible (for example, if a crash dump is collected and a document was in memory at the time of the crash). On the other hand, functional data can contain personal information. However, a user action, such as requesting news or asking Cortana a question, usually triggers collection and transmission of functional data.



If you're an IT pro that wants to manage Windows functional data sent from your organization to Microsoft, see [Manage connections from Windows operating system components to Microsoft services](#).

The following are specific examples of functional data:

- Current location for weather
- Bing searches
- Wallpaper and desktop settings synced across multiple devices

### **Diagnostic data gives users a voice**

Windows and Windows Server diagnostic data gives every user a voice in the operating system's development and ongoing improvement. It helps us understand how Windows 10 and Windows Server 2016 behaves in the real world, focus on user priorities, and make informed decisions that benefit them. For our enterprise customers, representation in the dataset on which we will make future design decisions is a real benefit. The following sections offer real examples of these benefits.

### **Drive higher app and driver quality**

Our ability to collect diagnostic data that drives improvements to Windows and Windows Server helps raise the bar for app and device driver quality. Diagnostic data helps us to quickly identify and fix critical reliability and security issues with apps and device drivers on given configurations. For example, we can identify an app that hangs on devices using a specific version of a video driver, allowing us to work with the app and device driver vendor to quickly fix the issue. The result is less downtime and reduced costs and increased productivity associated with troubleshooting these issues.

#### **Real-world example of how Windows diagnostic data helps**

There was a version of a video driver that was crashing on some devices running Windows 10, causing the device to reboot. We detected the problem in our diagnostic data, and immediately contacted the third-party developer who builds the video driver. Working with the developer, we provided an updated driver to Windows Insiders within 24 hours. Based on diagnostic data from the Windows Insiders' devices, we were able to validate the new version of the video driver, and rolled it out to the broad public as an update the next day. Diagnostic data helped us find, fix, and resolve this problem in just 48 hours, providing a better user experience and reducing costly support calls.

### **Improve end-user productivity**

Windows diagnostic data also helps Microsoft better understand how customers use (or do not use) the operating system's features and related services. The insights we gain from this data helps us prioritize our engineering effort to directly impact our customers' experiences. Examples are:

- **Start menu.** How do people change the Start menu layout? Do they pin other apps to it? Are there any apps that they frequently unpin? We use this dataset to adjust the default Start menu layout to better reflect people's expectations when they turn on their device for the first time.
- **Cortana.** We use diagnostic data to monitor the scalability of our cloud service, improving search performance.
- **Application switching.** Research and observations from earlier Windows versions showed that people rarely used Alt+Tab to switch between applications. After discussing this with some users, we learned they loved the feature, saying that it would be highly productive, but they did not know about it previously. Based on this, we created the Task View button in Windows 10 to make this feature more discoverable. Later diagnostic data showed significantly higher usage of this feature.

**These examples show how the use of diagnostic data enables Microsoft to build or enhance features which can help organizations increase employee productivity while lowering help desk calls.**

### **Insights into your own organization**

Sharing information with Microsoft helps make Windows and other products better, but it can also help make your internal processes and user experiences better, as well. Microsoft is in the process of developing a set of

analytics customized for your internal use. The first of these, called [Upgrade Readiness](#).

### **Upgrade Readiness**

Upgrading to new operating system versions has traditionally been a challenging, complex, and slow process for many enterprises. Discovering applications and drivers and then testing them for potential compatibility issues have been among the biggest pain points.

To better help customers through this difficult process, Microsoft developed Upgrade Readiness to give enterprises the tools to plan and manage the upgrade process end to end and allowing them to adopt new Windows releases more quickly and on an ongoing basis.

With Windows diagnostic data enabled, Microsoft collects computer, application, and driver compatibility-related information for analysis. We then identify compatibility issues that can block your upgrade and suggest fixes when they are known to Microsoft.

Use Upgrade Readiness to get:

- A visual workflow that guides you from pilot to production
- Detailed computer, driver, and application inventory
- Powerful computer level search and drill-downs
- Guidance and insights into application and driver compatibility issues with suggested fixes
- Data driven application rationalization tools
- Application usage information, allowing targeted validation; workflow to track validation progress and decisions
- Data export to commonly used software deployment tools

The Upgrade Readiness workflow steps you through the discovery and rationalization process until you have a list of computers that are ready to be upgraded.

## How is diagnostic data handled by Microsoft?

### **Data collection**

Windows 10 and Windows Server 2016 includes the Connected User Experiences and Telemetry component, which uses Event Tracing for Windows (ETW) tracelogging technology that gathers and stores diagnostic data events and data. The operating system and some Microsoft management solutions, such as System Center, use the same logging technology.

1. Operating system features and some management applications are instrumented to publish events and data. Examples of management applications include Virtual Machine Manager (VMM), Server Manager, and Storage Spaces.
2. Events are gathered using public operating system event logging and tracing APIs.
3. You can configure the diagnostic data level by using MDM policy, Group Policy, or registry settings.
4. The Connected User Experiences and Telemetry component transmits the diagnostic data.

Info collected at the Enhanced and Full levels of diagnostic data is typically gathered at a fractional sampling rate, which can be as low as 1% of devices reporting data at those levels.

### **Data transmission**

All diagnostic data is encrypted using SSL and uses certificate pinning during transfer from the device to the Microsoft Data Management Service. With Windows 10, data is uploaded on a schedule that is sensitive to event priority, battery use, and network cost. Real-time events, such as Windows Defender Advanced Threat Protection, are always sent immediately. Normal events are not uploaded on metered networks, unless you are on a metered server connection. On a free network, normal events can be uploaded every 4 hours if on battery, or every 15 minutes if on A/C power. Diagnostic and crash data are only uploaded on A/C power and free networks.

The data transmitted at the Basic and Enhanced data diagnostic levels is quite small; typically less than 1 MB per device per day, but occasionally up to 2 MB per device per day).

## Endpoints

The Microsoft Data Management Service routes data back to our secure cloud storage. Only Microsoft personnel with a valid business justification are permitted access.

The following table defines the endpoints for Connected User Experiences and Telemetry component:

WINDOWS RELEASE	ENDPOINT
Windows 10, versions 1703 and 1709	Diagnostics data: v10.vortex-win.data.microsoft.com/collect/v1 Functional: v20.vortex-win.data.microsoft.com/collect/v1 Windows Advanced Threat Protection is country specific and the prefix changes by country for example: <b>de</b> .vortex-win.data.microsoft.com/collect/v1 settings-win.data.microsoft.com
Windows 10, version 1607	v10.vortex-win.data.microsoft.com settings-win.data.microsoft.com

The following table defines the endpoints for other diagnostic data services:

SERVICE	ENDPOINT
<a href="#">Windows Error Reporting</a>	watson.telemetry.microsoft.com
<a href="#">Online Crash Analysis</a>	oca.telemetry.microsoft.com
OneDrive app for Windows 10	vortex.data.microsoft.com/collect/v1

## Data use and access

The principle of least privileged access guides access to diagnostic data. Microsoft does not share personal data of our customers with third parties, except at the customer's discretion or for the limited purposes described in the [Privacy Statement](#). Microsoft may share business reports with OEMs and third-party partners that include aggregated and anonymized diagnostic data information. Data-sharing decisions are made by an internal team including privacy, legal, and data management.

## Retention

Microsoft believes in and practices information minimization. We strive to gather only the info we need and to store it only for as long as it's needed to provide a service or for analysis. Much of the info about how Windows and apps are functioning is deleted within 30 days. Other info may be retained longer, such as error reporting data or Microsoft Store purchase history.

## Diagnostic data levels

This section explains the different diagnostic data levels in Windows 10, Windows Server 2016, and System Center. These levels are available on all desktop and mobile editions of Windows 10, except for the **Security** level, which is limited to Windows 10 Enterprise, Windows 10 Education, Windows 10 Mobile Enterprise, Windows 10 IoT Core (IoT Core), and Windows Server 2016.

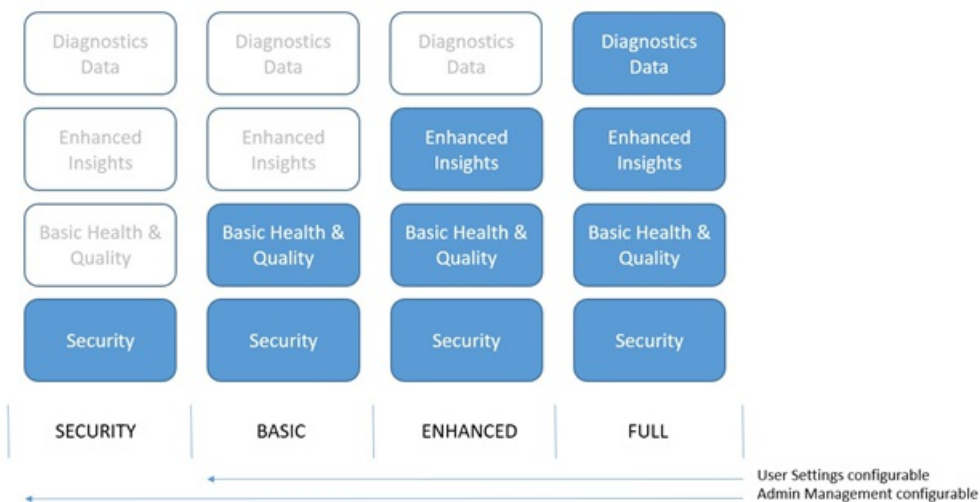
The diagnostic data is categorized into four levels:

- **Security.** Information that's required to help keep Windows, Windows Server, and System Center secure, including data about the Connected User Experiences and Telemetry component settings, the Malicious

Software Removal Tool, and Windows Defender.

- **Basic.** Basic device info, including: quality-related data, app compatibility, and data from the **Security** level.
- **Enhanced.** Additional insights, including: how Windows, Windows Server, System Center, and apps are used, how they perform, advanced reliability data, and data from both the **Basic** and the **Security** levels.
- **Full.** All data necessary to identify and help to fix problems, plus data from the **Security, Basic,** and **Enhanced** levels.

The levels are cumulative and are illustrated in the following diagram. Also, these levels apply to all editions of Windows Server 2016.



## Security level

The Security level gathers only the diagnostic data info that is required to keep Windows devices, Windows Server, and guests protected with the latest security updates. This level is only available on Windows Server 2016, Windows 10 Enterprise, Windows 10 Education, Windows 10 Mobile Enterprise, and Windows IoT Core editions.

### NOTE

If your organization relies on Windows Update for updates, you shouldn't use the **Security** level. Because no Windows Update information is gathered at this level, important information about update failures is not sent. Microsoft uses this information to fix the causes of those failures and improve the quality of our updates.

Windows Server Update Services (WSUS) and System Center Configuration Manager functionality is not affected at this level, nor is diagnostic data about Windows Server features or System Center gathered.

The data gathered at this level includes:

- **Connected User Experiences and Telemetry component settings.** If general diagnostic data has been gathered and is queued, it is sent to Microsoft. Along with this diagnostic data, the Connected User Experiences and Telemetry component may download a configuration settings file from Microsoft's servers. This file is used to configure the Connected User Experiences and Telemetry component itself. The data gathered by the client for this request includes OS information, device id (used to identify what specific device is requesting settings) and device class (for example, whether the device is server or desktop).
- **Malicious Software Removal Tool (MSRT)** The MSRT infection report contains information, including device info and IP address.

#### NOTE

You can turn off the MSRT infection report. No MSRT information is included if MSRT is not used. If Windows Update is turned off, MSRT will not be offered to users. For more info, see Microsoft KB article [891716](#).

- **Windows Defender/Endpoint Protection.** Windows Defender and System Center Endpoint Protection requires some information to function, including: anti-malware signatures, diagnostic information, User Account Control settings, Unified Extensible Firmware Interface (UEFI) settings, and IP address.

#### NOTE

This reporting can be turned off and no information is included if a customer is using third-party antimalware software, or if Windows Defender is turned off. For more info, see [Windows Defender](#).

Microsoft recommends that Windows Update, Windows Defender, and MSRT remain enabled unless the enterprise uses alternative solutions such as Windows Server Update Services, System Center Configuration Manager, or a third-party antimalware solution. Windows Update, Windows Defender, and MSRT provide core Windows functionality such as driver and OS updates, including security updates.

For servers with default diagnostic data settings and no Internet connectivity, you should set the diagnostic data level to **Security**. This stops data gathering for events that would not be uploaded due to the lack of Internet connectivity.

No user content, such as user files or communications, is gathered at the **Security** diagnostic data level, and we take steps to avoid gathering any information that directly identifies a company or user, such as name, email address, or account ID. However, in rare circumstances, MSRT information may unintentionally contain personal information. For instance, some malware may create entries in a computer's registry that include information such as a username, causing it to be gathered. MSRT reporting is optional and can be turned off at any time.

#### Basic level

The Basic level gathers a limited set of data that's critical for understanding the device and its configuration. This level also includes the **Security** level data. This level helps to identify problems that can occur on a specific hardware or software configuration. For example, it can help determine if crashes are more frequent on devices with a specific amount of memory or that are running a specific driver version. The Connected User Experiences and Telemetry component does not gather diagnostic data about System Center, but it can transmit diagnostic data for other non-Windows applications if they have user consent.

The normal upload range for the Basic diagnostic data level is between 109 KB - 159 KB per day, per device.

The data gathered at this level includes:

- **Basic device data.** Helps provide an understanding about the types of Windows devices and the configurations and types of native and virtualized Windows Server 2016 in the ecosystem. Examples include:
  - Device attributes, such as camera resolution and display type
  - Internet Explorer version
  - Battery attributes, such as capacity and type
  - Networking attributes, such as number of network adapters, speed of network adapters, mobile operator network, and IMEI number
  - Processor and memory attributes, such as number of cores, architecture, speed, memory size, and firmware

- Virtualization attribute, such as Second Level Address Translation (SLAT) support and guest operating system
- Operating system attributes, such as Windows edition and virtualization state
- Storage attributes, such as number of drives, type, and size
- **Connected User Experiences and Telemetry component quality metrics.** Helps provide an understanding about how the Connected User Experiences and Telemetry component is functioning, including % of uploaded events, dropped events, and the last upload time.
- **Quality-related information.** Helps Microsoft develop a basic understanding of how a device and its operating system are performing. Some examples are the device characteristics of a Connected Standby device, the number of crashes or hangs, and application state change details, such as how much processor time and memory were used, and the total uptime for an app.
- **Compatibility data.** Helps provide an understanding about which apps are installed on a device or virtual machine and identifies potential compatibility problems.
  - **General app data and app data for Internet Explorer add-ons.** Includes a list of apps that are installed on a native or virtualized instance of the OS and whether these apps function correctly after an upgrade. This app data includes the app name, publisher, version, and basic details about which files have been blocked from usage.
  - **Internet Explorer add-ons.** Includes a list of Internet Explorer add-ons that are installed on a device and whether these apps will work after an upgrade.
  - **System data.** Helps provide an understanding about whether a device meets the minimum requirements to upgrade to the next version of the operating system. System information includes the amount of memory, as well as information about the processor and BIOS.
  - **Accessory device data.** Includes a list of accessory devices, such as printers or external storage devices, that are connected to Windows PCs and whether these devices will function after upgrading to a new version of the operating system.
  - **Driver data.** Includes specific driver usage that's meant to help figure out whether apps and devices will function after upgrading to a new version of the operating system. This can help to determine blocking issues and then help Microsoft and our partners apply fixes and improvements.
- **Microsoft Store.** Provides information about how the Microsoft Store performs, including app downloads, installations, and updates. It also includes Microsoft Store launches, page views, suspend and resumes, and obtaining licenses.

### Enhanced level

The Enhanced level gathers data about how Windows and apps are used and how they perform. This level also includes data from both the **Basic** and **Security** levels. This level helps to improve the user experience with the operating system and apps. Data from this level can be abstracted into patterns and trends that can help Microsoft determine future improvements.

This is the default level for Windows 10 Enterprise and Windows 10 Education editions, and the minimum level needed to quickly identify and address Windows, Windows Server, and System Center quality issues.

The normal upload range for the Enhanced diagnostic data level is between 239 KB - 348 KB per day, per device.

The data gathered at this level includes:

- **Operating system events.** Helps to gain insights into different areas of the operating system, including networking, Hyper-V, Cortana, storage, file system, and other components.

- **Operating system app events.** A set of events resulting from Microsoft applications and management tools that were downloaded from the Store or pre-installed with Windows or Windows Server, including Server Manager, Photos, Mail, and Microsoft Edge.
- **Device-specific events.** Contains data about events that are specific to certain devices, such as Surface Hub and Microsoft HoloLens. For example, Microsoft HoloLens sends Holographic Processing Unit (HPU)-related events.
- **Some crash dump types.** All crash dump types, except for heap dumps and full dumps.

If the Connected User Experiences and Telemetry component detects a problem on Windows 10 that requires gathering more detailed instrumentation, the Connected User Experiences and Telemetry component at the **Enhanced** diagnostic data level will only gather data about the events associated with the specific issue.

#### **Limit Enhanced diagnostic data to the minimum required by Windows Analytics**

Windows Analytics Device Health reports are powered by diagnostic data not included in the **Basic** level, such as crash reports and certain operating system events. In the past, organizations sending **Enhanced** or **Full** level diagnostic data were able to participate in Device Health. However, organizations that required detailed event and field level documentation were unable to move from **Basic** to **Enhanced**.

In Windows 10, version 1709, we introduce the **Limit Enhanced diagnostic data to the minimum required by Windows Analytics** feature. When enabled, this feature lets you send only the following subset of **Enhanced** level diagnostic data. For more info about Device Health, see the [Monitor the health of devices with Device Health](#) topic.

- **Operating system events.** Limited to a small set required for analytics reports and documented in the [Windows 10, version 1709 enhanced diagnostic data events and fields used by Windows Analytics](#) topic.
- **Some crash dump types.** All crash dump types, except for heap and full dumps.

#### **To turn on this behavior for devices**

1. Set the diagnostic data level to **Enhanced**, using either Group Policy or MDM.
  - a. Using Group Policy, set the **Computer Configuration/Administrative Templates/Windows Components/Data Collection and Preview Builds/Allow telemetry** setting to **2**.

-OR-

  - b. Using MDM, use the Policy CSP to set the **System/AllowTelemetry** value to **2**.

-AND-
2. Enable the **LimitEnhancedDiagnosticDataWindowsAnalytics** setting, using either Group Policy or MDM.
  - a. Using Group Policy, set the **Computer Configuration/Administrative Templates/Windows Components/Data collection and Preview builds/Limit Enhanced diagnostic data to the minimum required by Windows Analytics** setting to **Enabled**.

-OR-

  - b. Using MDM, use the Policy CSP to set the **System/LimitEnhancedDiagnosticDataWindowsAnalytics** value to **1**.

#### **Full level**

The **Full** level gathers data necessary to identify and to help fix problems, following the approval process described below. This level also includes data from the **Basic**, **Enhanced**, and **Security** levels. This is the default level for Windows 10 Pro.

Additionally, at this level, devices opted in to the [Windows Insider Program](#) will send events, such as reliability and app responsiveness. that can show Microsoft how pre-release binaries and features are performing. These events help us make decisions on which builds are flighted. All devices in the [Windows Insider Program](#) are automatically set to this level.

If a device experiences problems that are difficult to identify or repeat using Microsoft's internal testing, additional data becomes necessary. This data can include any user content that might have triggered the problem and is gathered from a small sample of devices that have both opted into the **Full** diagnostic data level and have exhibited the problem.

However, before more data is gathered, Microsoft's privacy governance team, including privacy and other subject matter experts, must approve the diagnostics request made by a Microsoft engineer. If the request is approved, Microsoft engineers can use the following capabilities to get the information:

- Ability to run a limited, pre-approved list of Microsoft certified diagnostic tools, such as msinfo32.exe, powercfg.exe, and dxdiag.exe.
- Ability to get registry keys.
- All crash dump types, including heap dumps and full dumps.

## Enterprise management

Sharing diagnostic data with Microsoft provides many benefits to enterprises, so we do not recommend turning it off. For most enterprise customers, simply adjusting the diagnostic data level and managing specific components is the best option.

Customers can set the diagnostic data level in both the user interface and with existing management tools. Users can change the diagnostic data level in the **Diagnostic data** setting. In the **Settings** app, it is in **Privacy\Feedback & diagnostics**. They can choose between Basic and Full. The Enhanced level will only be displayed as an option when Group Policy or Mobile Device Management (MDM) are invoked with this level. The Security level is not available.

IT pros can use various methods, including Group Policy and Mobile Device Management (MDM), to choose a diagnostic data level. If you're using Windows 10 Enterprise, Windows 10 Education, or Windows Server 2016, the Security diagnostic data level is available when managing the policy. Setting the diagnostic data level through policy sets the upper boundary for the users' choices. To disable user choice after setting the level with the policy, you will need to use the "Configure telemetry opt-in setting user interface" group policy. The remainder of this section describes how to use group policy to configure levels and settings interface.

### Manage your diagnostic data settings

We do not recommend that you turn off diagnostic data in your organization as valuable functionality may be impacted, but we recognize that in some scenarios this may be required. Use the steps in this section to do so for Windows, Windows Server, and System Center.

#### IMPORTANT

These diagnostic data levels only apply to Windows, Windows Server, and System Center components and apps that use the Connected User Experiences and Telemetry component. Non-Windows components, such as Microsoft Office or other 3rd-party apps, may communicate with their cloud services outside of these diagnostic data levels. You should work with your app vendors to understand their diagnostic data policy, and how you can to opt in or opt out. For more information on how Microsoft Office uses diagnostic data, see [Overview of Office Telemetry](#).

You can turn on or turn off System Center diagnostic data gathering. The default is on and the data gathered at this level represents what is gathered by default when System Center diagnostic data is turned on. However, setting the operating system diagnostic data level to **Basic** will turn off System Center diagnostic data, even if the



System Center diagnostic data switch is turned on.

The lowest diagnostic data setting level supported through management policies is **Security**. The lowest diagnostic data setting supported through the Settings UI is **Basic**. The default diagnostic data setting for Windows Server 2016 is **Enhanced**.

### Configure the operating system diagnostic data level

You can configure your operating system diagnostic data settings using the management tools you're already using, such as Group Policy, MDM, or Windows Provisioning. You can also manually change your settings using Registry Editor. Setting your diagnostic data levels through a management policy sets the upper level for diagnostic data on the device.

Use the appropriate value in the table below when you configure the management policy.

LEVEL	DATA GATHERED	VALUE
Security	Security data only.	0
Basic	Security data, and basic system and quality data.	1
Enhanced	Security data, basic system and quality data, and enhanced insights and advanced reliability data.	2
Full	Security data, basic system and quality data, enhanced insights and advanced reliability data, and full diagnostics data.	3

#### NOTE

When the User Configuration policy is set for Diagnostic Data, this will override the Computer Configuration setting.

### Use Group Policy to set the diagnostic data level

Use a Group Policy object to set your organization's diagnostic data level.

1. From the Group Policy Management Console, go to **Computer Configuration > Administrative Templates > Windows Components > Data Collection and Preview Builds**.
2. Double-click **Allow Telemetry**.
3. In the **Options** box, select the level that you want to configure, and then click **OK**.

### Use MDM to set the diagnostic data level

Use the [Policy Configuration Service Provider \(CSP\)](#) to apply the System/AllowTelemetry MDM policy.

### Use Registry Editor to set the diagnostic data level

Use Registry Editor to manually set the registry level on each device in your organization or you can write a script to edit the registry. If a management policy already exists, such as Group Policy or MDM, it will override this registry setting.

1. Open Registry Editor, and go to **HKEY\_LOCAL\_MACHINE\Software\Microsoft\Windows\CurrentVersion\Policies\DataCollection**.
2. Right-click **DataCollection**, click New, and then click **DWORD (32-bit) Value**.

3. Type **AllowTelemetry**, and then press ENTER.
4. Double-click **AllowTelemetry**, set the desired value from the table above, and then click **OK**.
5. Click **File** > **Export**, and then save the file as a .reg file, such as **C:\AllowTelemetry.reg**. You can run this file from a script on each device in your organization.

### Configure System Center 2016 diagnostic data

For System Center 2016 Technical Preview, you can turn off System Center diagnostic data by following these steps:

- Turn off diagnostic data by using the System Center UI Console settings workspace.
- For information about turning off diagnostic data for Service Management Automation and Service Provider Foundation, see [How to disable telemetry for Service Management Automation and Service Provider Foundation](#).

### Additional diagnostic data controls

There are a few more settings that you can turn off that may send diagnostic data information:

- To turn off Windows Update diagnostic data, you have two choices. Either turn off Windows Update, or set your devices to be managed by an on premises update server, such as [Windows Server Update Services \(WSUS\)](#) or [System Center Configuration Manager](#).
- Turn off **Windows Defender Cloud-based Protection** and **Automatic sample submission** in **Settings > Update & security > Windows Defender**.
- Manage the Malicious Software Removal Tool in your organization. For more info, see Microsoft KB article [891716](#).
- Turn off **Linguistic Data Collection** in **Settings > Privacy**. At diagnostic data levels **Enhanced** and **Full**, Microsoft uses Linguistic Data Collection info to improve language model features such as autocomplete, spellcheck, suggestions, input pattern recognition, and dictionary.

#### NOTE

Microsoft does not intend to gather sensitive information, such as credit card numbers, usernames and passwords, email addresses, or other similarly sensitive information for Linguistic Data Collection. We guard against such events by using technologies to identify and remove sensitive information before linguistic data is sent from the user's device. If we determine that sensitive information has been inadvertently received, we delete the information.

## Additional resources

### FAQs

- [Cortana, Search, and privacy](#)
- [Windows 10 feedback, diagnostics, and privacy](#)
- [Windows 10 camera and privacy](#)
- [Windows 10 location service and privacy](#)
- [Microsoft Edge and privacy](#)
- [Windows 10 speech, inking, typing, and privacy](#)
- [Windows Hello and privacy](#)
- [Wi-Fi Sense](#)
- [Windows Update Delivery Optimization](#)

## Blogs

- [Privacy and Windows 10](#)

## Privacy Statement

- [Microsoft Privacy Statement](#)

## TechNet

- [Manage connections from Windows operating system components to Microsoft services](#)

## Web Pages

- [Privacy at Microsoft](#)

# Diagnostic Data Viewer Overview

5/22/2018 • 4 minutes to read • [Edit Online](#)

## Applies to

- Windows 10, version 1803

## Introduction

The Diagnostic Data Viewer is a Windows app that lets you review the diagnostic data your device is sending to Microsoft, grouping the info into simple categories based on how it's used by Microsoft.

## Install and Use the Diagnostic Data Viewer

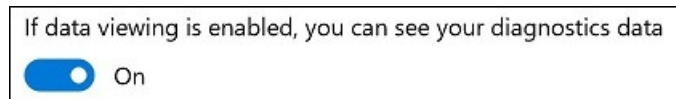
You must turn on data viewing and download the app before you can use the Diagnostic Data Viewer to review your device's diagnostic data.

### Turn on data viewing

Before you can use this tool, you must turn on data viewing in the **Settings** panel. Turning on data viewing lets Windows store your device's diagnostic data until you turn it off. Turning off data viewing stops Windows from collecting your diagnostic data and clears the existing diagnostic data from your device.

### To turn on data viewing

1. Go to **Start**, select **Settings** > **Privacy** > **Diagnostics & feedback**.
2. Under **Diagnostic data**, turn on the **If data viewing is enabled, you can see your diagnostics data** option.



### Download the Diagnostic Data Viewer

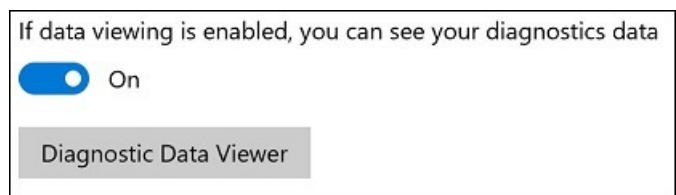
Download the app from the [Microsoft Store Diagnostic Data Viewer](#) page.

### Start the Diagnostic Data Viewer

You must start this app from the **Settings** panel.

### To start the Diagnostic Data Viewer

1. Go to **Start**, select **Settings** > **Privacy** > **Diagnostics & feedback**.
2. Under **Diagnostic data**, select the **Diagnostic Data Viewer** button.



-OR-

Go to **Start** and search for *Diagnostic Data Viewer*.

3. Close the Diagnostic Data Viewer app, use your device as you normally would for a few days, and then open Diagnostic Data Viewer again to review the updated list of diagnostic data.

#### IMPORTANT

Turning on data viewing can use up to 1GB of disk space on your system drive. We strongly recommend that you turn off data viewing when you're done using the Diagnostic Data Viewer. For info about turning off data viewing, see the [Turn off data viewing](#) section in this article.

### Use the Diagnostic Data Viewer

The Diagnostic Data Viewer provides you with the following features to view and filter your device's diagnostic data.

- **View your diagnostic events.** In the left column, you can review your diagnostic events. These events reflect activities that occurred and were sent to Microsoft.

Selecting an event opens the detailed JSON view, which provides the exact details uploaded to Microsoft. Microsoft uses this info to continually improve the Windows operating system.



- **Search your diagnostic events.** The **Search** box at the top of the screen lets you search amongst all of the diagnostic event details. The returned search results include any diagnostic event that contains the matching text.

Selecting an event opens the detailed JSON view, with the matching text highlighted.

- **Filter your diagnostic event categories.** The apps Menu button opens the detailed menu. In here, you'll find a list of diagnostic event categories, which define how the events are used by Microsoft.

Selecting a check box lets you filter between the diagnostic event categories.

- **Help to make your Windows experience better.** Microsoft samples diagnostic data from a small amount of devices to make big improvements to the Windows operating system and ultimately, your experience. If you're a part of this small device group and you experience issues, Microsoft will collect the associated event diagnostic data, allowing your info to potentially help fix the issue for others.

To signify your contribution, you'll see this icon () if your device is part of the sampling group. In addition, if any of your diagnostic data events are sent from your device to Microsoft to help make improvements, you'll see this icon (.

- **Provide diagnostic event feedback.** The **Feedback** icon opens the Feedback Hub app, letting you provide feedback about the Diagnostic Data Viewer and the diagnostic events.

Selecting a specific event in the Diagnostic Data Viewer automatically fills in the field in the Feedback Hub. You can add your comments to the box labeled, **Give us more detail (optional)**.

#### IMPORTANT

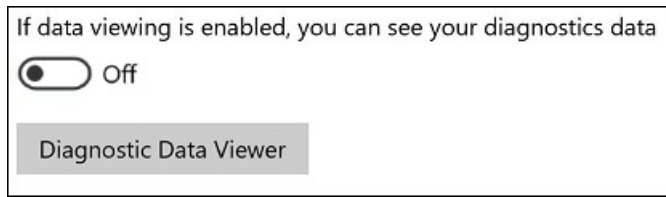
All content in the Feedback Hub is publicly viewable. Therefore, make sure you don't put any personal info into your feedback comments.

## Turn off data viewing

When you're done reviewing your diagnostic data, you should turn off data viewing.

## To turn off data viewing

1. Go to **Start**, select **Settings > Privacy > Diagnostics & feedback**.
2. Under **Diagnostic data**, turn off the **If data viewing is enabled, you can see your diagnostics data** option.



## View additional diagnostic data in the View problem reports tool

You can review additional Windows Error Reporting diagnostic data in the **View problem reports** tool. This tool provides you with a summary of various crash reports that are sent to Microsoft as part of Windows Error Reporting. We use this data to find and fix specific issues that are hard to replicate and to improve the Windows operating system.

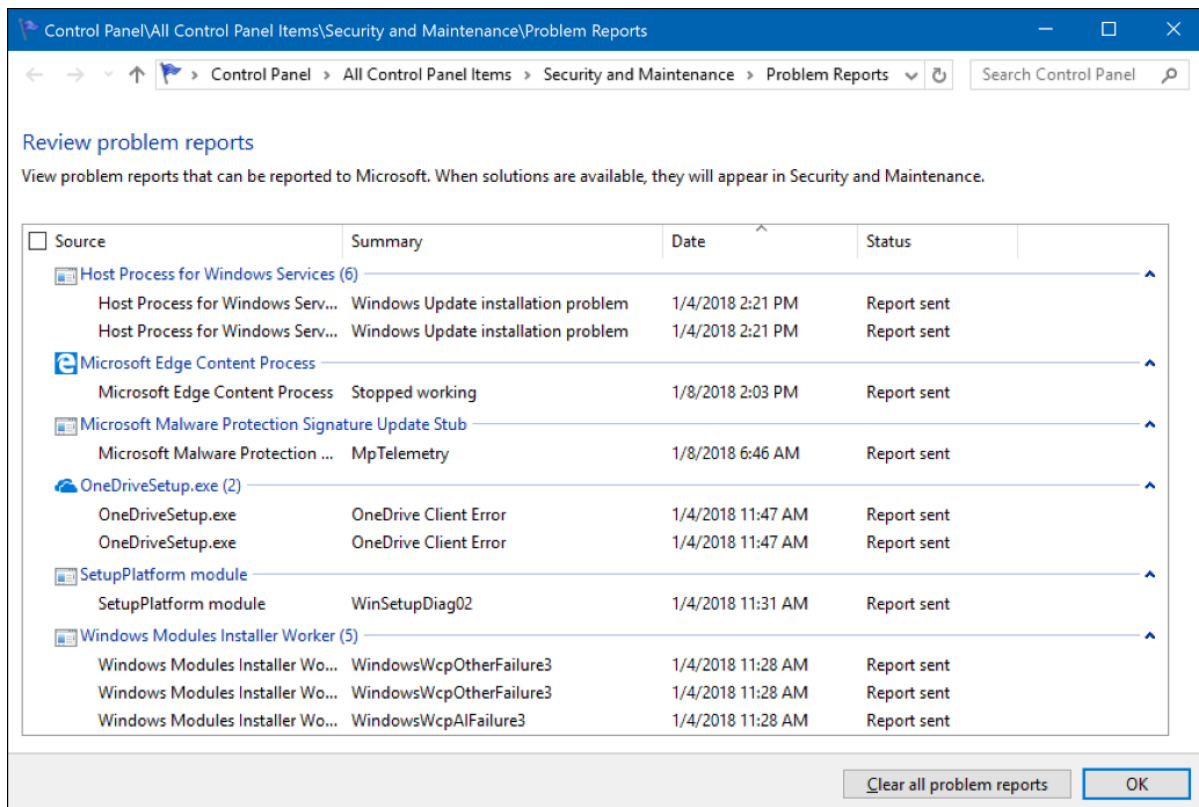
### To view your Windows Error Reporting diagnostic data

1. Go to **Start**, select **Control Panel > All Control Panel Items > Security and Maintenance > Problem Reports**.

-OR-

Go to **Start** and search for *Problem Reports*.

The **Review problem reports** tool opens, showing you your Windows Error Reporting reports, along with a status about whether it was sent to Microsoft.



# Windows 10, version 1803 basic level Windows diagnostic events and fields

7/9/2018 • 177 minutes to read • [Edit Online](#)

## Applies to

- Windows 10, version 1803

The Basic level gathers a limited set of information that is critical for understanding the device and its configuration including: basic device information, quality-related information, app compatibility, and Microsoft Store. When the level is set to Basic, it also includes the Security level information.

The Basic level helps to identify problems that can occur on a particular device hardware or software configuration. For example, it can help determine if crashes are more frequent on devices with a specific amount of memory or that are running a particular driver version. This helps Microsoft fix operating system or app problems.

Use this article to learn about diagnostic events, grouped by event area, and the fields within each event. A brief description is provided for each field. Every event generated includes common data, which collects device data.

You can learn more about Windows functional and diagnostic data through these articles:

- [Windows 10, version 1709 basic diagnostic events and fields](#)
- [Windows 10, version 1703 basic diagnostic events and fields](#)
- [Manage connections from Windows operating system components to Microsoft services](#)
- [Manage Windows 10 connection endpoints](#)
- [Configure Windows diagnostic data in your organization](#)

## Common data extensions

### Common Data Extensions.App

The following fields are available:

- **expld** Associates a flight, such as an OS flight, or an experiment, such as a web site UX experiment, with an event.
- **userId** The userID as known by the application.
- **env** The environment from which the event was logged.
- **asId** An integer value that represents the app session. This value starts at 0 on the first app launch and increments after each subsequent app launch per boot session.
- **id** Represents a unique identifier of the client application currently loaded in the process producing the event; and is used to group events together and understand usage pattern, errors by application.
- **ver** Represents the version number of the application. Used to understand errors by Version, Usage by Version across an app.

### Common Data Extensions.CS

The following fields are available:

- **sig** A common schema signature that identifies new and modified event schemas.

### Common Data Extensions.CUET

The following fields are available:

- **stId** Represents the Scenario Entry Point ID. This is a unique GUID for each event in a diagnostic scenario. This used to be Scenario Trigger ID.
- **ald** Represents the ETW ActivityId. Logged via TraceLogging or directly via ETW.
- **rald** Represents the ETW Related ActivityId. Logged via TraceLogging or directly via ETW.
- **op** Represents the ETW Op Code.
- **cat** Represents a bitmask of the ETW Keywords associated with the event.
- **flags** Represents the bitmap that captures various Windows specific flags.
- **cpId** The composer ID, such as Reference, Desktop, Phone, Holographic, Hub, IoT Composer.
- **tickets** A list of strings that represent entries in the HTTP header of the web request that includes this event.
- **bseq** Upload buffer sequence number in the format <buffer identifier>:<sequence number>
- **mon** Combined monitor and event sequence numbers in the format <monitor sequence>:<event sequence>
- **epoch** Represents the epoch and seqNum fields, which help track how many events were fired and how many events were uploaded, and enables identification of data lost during upload and de-duplication of events on the ingress server.
- **seq** Represents the sequence field used to track absolute order of uploaded events. It is an incrementing identifier for each event added to the upload queue. The Sequence helps track how many events were fired and how many events were uploaded and enables identification of data lost during upload and de-duplication of events on the ingress server.

#### Common Data Extensions.Device

The following fields are available:

- **ver** Represents the major and minor version of the extension.
- **localId** Represents a locally defined unique ID for the device, not the human readable device name. Most likely equal to the value stored at HKLM\Software\Microsoft\SQMClient\MachineId
- **deviceClass** Represents the classification of the device, the device "family". For example, Desktop, Server, or Mobile.

#### Common Data Extensions.Envelope

The following fields are available:

- **ver** Represents the major and minor version of the extension.
- **name** Represents the uniquely qualified name for the event.
- **time** Represents the event date time in Coordinated Universal Time (UTC) when the event was generated on the client. This should be in ISO 8601 format.
- **popSample** Represents the effective sample rate for this event at the time it was generated by a client.
- **iKey** Represents an ID for applications or other logical groupings of events.
- **flags** Represents a collection of bits that describe how the event should be processed by the Connected User Experience and Telemetry component pipeline. The lowest-order byte is the event persistence. The next byte is the event latency.
- **cV** Represents the Correlation Vector: A single field for tracking partial order of related telemetry events across component boundaries.

#### Common Data Extensions.OS

The following fields are available:

- **ver** Represents the major and minor version of the extension.
- **expId** Represents the experiment ID. The standard for associating a flight, such as an OS flight (pre-release build), or an experiment, such as a web site UX experiment, with an event is to record the flight / experiment IDs in Part A of the common schema.
- **locale** Represents the locale of the operating system.



- **bootId** An integer value that represents the boot session. This value starts at 0 on first boot after OS install and increments after every reboot.
- **os** Represents the operating system name.
- **ver** Represents the OS version, and its format is OS dependent.

#### Common Data Extensions.User

The following fields are available:

- **ver** Represents the major and minor version of the extension.
- **localId** Represents a unique user identity that is created locally and added by the client. This is not the user's account ID.

#### Common Data Extensions.XBL

The following fields are available:

- **nbf** Not before time
- **expId** Expiration time
- **sbx** XBOX sandbox identifier
- **dtv** XBOX device type
- **did** XBOX device ID
- **xid** A list of base10-encoded XBOX User IDs.
- **uts** A bit field, with 2 bits being assigned to each user ID listed in xid. This field is omitted if all users are retail accounts.

#### Common Data Extensions.Consent UI Event

This User Account Control (UAC) telemetry point collects information on elevations that originate from low integrity levels. This occurs when a process running at low integrity level (IL) requires higher (administrator) privileges, and therefore requests for elevation via UAC (consent.exe). By better understanding the processes requesting these elevations, Microsoft can in turn improve the detection and handling of potentially malicious behavior in this path.

The following fields are available:

- **eventType** Represents the type of elevation: If it succeeded, was cancelled, or was auto-approved.
- **splitToken** Represents the flag used to distinguish between administrators and standard users.
- **friendlyName** Represents the name of the file requesting elevation from low IL.
- **elevationReason** Represents the distinction between various elevation requests sources (appcompat, installer, COM, MSI and so on).
- **exeName** Represents the name of the file requesting elevation from low IL.
- **signatureState** Represents the state of the signature, if it signed, unsigned, OS signed and so on.
- **publisherName** Represents the name of the publisher of the file requesting elevation from low IL.
- **cmdLine** Represents the full command line arguments being used to elevate.
- **Hash.Length** Represents the length of the hash of the file requesting elevation from low IL.
- **Hash** Represents the hash of the file requesting elevation from low IL.
- **HashAlgId** Represents the algorithm ID of the hash of the file requesting elevation from low IL.
- **telemetryFlags** Represents the details about the elevation prompt for CEIP data.
- **timeStamp** Represents the time stamp on the file requesting elevation.
- **fileVersionMS** Represents the major version of the file requesting elevation.
- **fileVersionLS** Represents the minor version of the file requesting elevation.

## Common data fields

## Common Data Fields.MS.Device.DeviceInventory.Change

These fields are added whenever Ms.Device.DeviceInventoryChange is included in the event.

The following fields are available:

- **syncId** A string used to group StartSync, EndSync, Add, and Remove operations that belong together. This field is unique by Sync period and is used to disambiguate in situations where multiple agents perform overlapping inventories for the same object.
- **objectType** Indicates the object type that the event applies to.
- **Action** The change that was invoked on a device inventory object.
- **inventoryId** Device ID used for Compatibility testing

## Common Data Fields.TelClientSynthetic.PrivacySettingsAfterCreatorsUpdate.PreUpgradeSettings

These fields are added whenever PreUpgradeSettings is included in the event.

The following fields are available:

- **HKLM\_SensorPermissionState.SensorPermissionState** The state of the Location service before the feature update completed.
- **HKLM\_SensorPermissionState.HRESULT** The error code returned when trying to query the Location service for the device.
- **HKCU\_SensorPermissionState.SensorPermissionState** The state of the Location service when a user signs on before the feature update completed.
- **HKCU\_SensorPermissionState.HRESULT** The error code returned when trying to query the Location service for the current user.
- **HKLM\_LocationPlatform.Status** The state of the location platform after the feature update has completed.
- **HKLM\_LocationPlatform.HRESULT** The error code returned when trying to query the location platform for the device.
- **HKLM\_LocationSyncEnabled.AcceptedPrivacyPolicy** The speech recognition state for the device before the feature update completed.
- **HKLM\_LocationSyncEnabled.HRESULT** The error code returned when trying to query the Find My Device service for the device.
- **HKCU\_LocationSyncEnabled.AcceptedPrivacyPolicy** The speech recognition state for the current user before the feature update completed.
- **HKCU\_LocationSyncEnabled.HRESULT** The error code returned when trying to query the Find My Device service for the current user.
- **HKLM\_AllowTelemetry.AllowTelemetry** The state of the Connected User Experiences and Telemetry component for the device before the feature update.
- **HKLM\_AllowTelemetry.HRESULT** The error code returned when trying to query the Connected User Experiences and Telemetry component for the device.
- **HKLM\_TIPC.Enabled** The state of TIPC for the device.
- **HKLM\_TIPC.HRESULT** The error code returned when trying to query TIPC for the device.
- **HKCU\_TIPC.Enabled** The state of TIPC for the current user.
- **HKCU\_TIPC.HRESULT** The error code returned when trying to query TIPC for the current user.
- **HKLM\_FlipAhead.FPEnabled** Is Flip Ahead enabled for the device before the feature update was completed?
- **HKLM\_FlipAhead.HRESULT** The error code returned when trying to query Flip Ahead for the device.
- **HKCU\_FlipAhead.FPEnabled** Is Flip Ahead enabled for the current user before the feature update was completed?
- **HKCU\_FlipAhead.HRESULT** The error code returned when trying to query Flip Ahead for the current user.
- **HKLM\_TailoredExperiences.TailoredExperiencesWithDiagnosticDataEnabled** Is Tailored Experiences with Diagnostics Data enabled for the current user after the feature update had completed?

- **HKCU\_TailoredExperiences.HRESULT** The error code returned when trying to query Tailored Experiences with Diagnostics Data for the current user.
- **HKLM\_AdvertisingID.Enabled** Is the advertising ID enabled for the device?
- **HKLM\_AdvertisingID.HRESULT** The error code returned when trying to query the state of the advertising ID for the device.
- **HKCU\_AdvertisingID.Enabled** Is the advertising ID enabled for the current user?
- **HKCU\_AdvertisingID.HRESULT** The error code returned when trying to query the state of the advertising ID for the user.

### **Common Data Fields.TelClientSynthetic.PrivacySettingsAfterCreatorsUpdate.PostUpgradeSettings**

These fields are added whenever PostUpgradeSettings is included in the event.

The following fields are available:

- **HKLM\_SensorPermissionState.SensorPermissionState** The state of the Location service after the feature update has completed.
- **HKLM\_SensorPermissionState.HRESULT** The error code returned when trying to query the Location service for the device.
- **HKCU\_SensorPermissionState.SensorPermissionState** The state of the Location service when a user signs on after a feature update has completed.
- **HKCU\_SensorPermissionState.HRESULT** The error code returned when trying to query the Location service for the current user.
- **HKLM\_LocationPlatform.Status** The state of the location platform after the feature update has completed.
- **HKLM\_LocationPlatform.HRESULT** The error code returned when trying to query the location platform for the device.
- **HKLM\_LocationSyncEnabled.AcceptedPrivacyPolicy** The speech recognition state for the device after the feature update has completed.
- **HKLM\_LocationSyncEnabled.HRESULT** The error code returned when trying to query the Find My Device service for the device.
- **HKCU\_LocationSyncEnabled.AcceptedPrivacyPolicy** The speech recognition state for the current user after the feature update has completed.
- **HKCU\_LocationSyncEnabled.HRESULT** The error code returned when trying to query the Find My Device service for the current user.
- **HKLM\_AllowTelemetry.AllowTelemetry** The state of the Connected User Experiences and Telemetry component for the device after the feature update.
- **HKLM\_AllowTelemetry.HRESULT** The error code returned when trying to query the Connected User Experiences and Telemetry component for the device.
- **HKLM\_TIPC.Enabled** The state of TIPC for the device.
- **HKLM\_TIPC.HRESULT** The error code returned when trying to query TIPC for the device.
- **HKCU\_TIPC.Enabled** The state of TIPC for the current user.
- **HKCU\_TIPC.HRESULT** The error code returned when trying to query TIPC for the current user.
- **HKLM\_FlipAhead.FPEnabled** Is Flip Ahead enabled for the device after the feature update has completed?
- **HKLM\_FlipAhead.HRESULT** The error code returned when trying to query Flip Ahead for the device.
- **HKCU\_FlipAhead.FPEnabled** Is Flip Ahead enabled for the current user after the feature update has completed?
- **HKCU\_FlipAhead.HRESULT** The error code returned when trying to query Flip Ahead for the current user.
- **HKLM\_TailoredExperiences.TailoredExperiencesWithDiagnosticDataEnabled** Is Tailored Experiences with Diagnostics Data enabled for the current user after the feature update had completed?
- **HKCU\_TailoredExperiences.HRESULT** The error code returned when trying to query Tailored Experiences with Diagnostics Data for the current user.

- **HKLM\_AdvertisingID.Enabled** Is the advertising ID enabled for the device?
- **HKLM\_AdvertisingID.HRESULT** The error code returned when trying to query the state of the advertising ID for the device.
- **HKCU\_AdvertisingID.Enabled** Is the advertising ID enabled for the current user?
- **HKCU\_AdvertisingID.HRESULT** The error code returned when trying to query the state of the advertising ID for the user.

## Appraiser events

### Microsoft.Windows.Appraiser.General.ChecksumTotalPictureCount

This event lists the types of objects and how many of each exist on the client device. This allows for a quick way to ensure that the records present on the server match what is present on the client.

The following fields are available:

- **PCFP** An ID for the system, calculated by hashing hardware identifiers.
- **SystemProcessorLahfSahf** The count of the number of this particular object type present on this device.
- **SystemProcessorCompareExchange** The count of the number of this particular object type present on this device.
- **SystemProcessorSse2** The count of the number of this particular object type present on this device.
- **SystemProcessorNx** The count of the number of this particular object type present on this device.
- **SystemWim** The count of the number of this particular object type present on this device.
- **SystemWlan** The count of the number of this particular object type present on this device.
- **DatasourceDevicePnp\_RS1** The total DataSourceDevicePnp objects targeting Windows 10 version 1607 on this device.
- **DecisionDevicePnp\_RS1** The total DecisionDevicePnp objects targeting Windows 10 version 1607 on this device.
- **InventorySystemBios** The count of the number of this particular object type present on this device.
- **DataSourceMatchingInfoPostUpgrade\_RS1** The total DataSourceMatchingInfoPostUpgrade objects targeting Windows 10 version 1607 on this device.
- **DecisionMatchingInfoPostUpgrade\_RS1** The total DecisionMatchingInfoPostUpgrade objects targeting Windows 10 version 1607 on this device.
- **SystemMemory** The count of the number of this particular object type present on this device.
- **SystemProcessorPrefetchW** The count of the number of this particular object type present on this device.
- **DatasourceSystemBios\_RS1** The total DatasourceSystemBios objects targeting Windows 10 version 1607 present on this device.
- **DecisionSystemBios\_RS1** The total DecisionSystemBios objects targeting Windows 10 version 1607 on this device.
- **DataSourceMatchingInfoPassive\_RS1** The total DataSourceMatchingInfoPassive objects targeting Windows 10 version 1607 on this device.
- **DecisionMatchingInfoPassive\_RS1** The total DecisionMatchingInfoPassive objects targeting Windows 10 version 1607 on this device.
- **InventoryUplevelDriverPackage** The count of the number of this particular object type present on this device.
- **DatasourceDriverPackage\_RS1** The total DataSourceDriverPackage objects targeting Windows 10 version 1607 on this device.
- **DecisionDriverPackage\_RS1** The total DecisionDriverPackage objects targeting Windows 10 version 1607 on this device.
- **Wmdrm\_RS1** An ID for the system, calculated by hashing hardware identifiers.
- **DecisionTest\_RS1** An ID for the system, calculated by hashing hardware identifiers.

- **SystemWindowsActivationStatus** The count of the number of this particular object type present on this device.
- **SystemTouch** The count of the number of this particular object type present on this device.
- **InventoryApplicationFile** The count of the number of this particular object type present on this device.
- **InventoryLanguagePack** The count of InventoryLanguagePack objects present on this machine.
- **InventoryMediaCenter** The count of the number of this particular object type present on this device.
- **DatasourceSystemBios\_RS3** The total DatasourceSystemBios objects targeting the next release of Windows on this device.
- **DecisionSystemBios\_RS3** The total DecisionSystemBios objects targeting the next release of Windows on this device.
- **DatasourceApplicationFile\_RS3** The total DecisionApplicationFile objects targeting the next release of Windows on this device.
- **DatasourceDevicePnp\_RS3** The total DatasourceDevicePnp objects targeting the next release of Windows on this device.
- **DatasourceDriverPackage\_RS3** The total DatasourceDriverPackage objects targeting the next release of Windows on this device.
- **DataSourceMatchingInfoBlock\_RS3** The total DataSourceMatchingInfoBlock objects targeting the next release of Windows on this device.
- **DataSourceMatchingInfoPassive\_RS3** The total DataSourceMatchingInfoPassive objects targeting the next release of Windows on this device.
- **DataSourceMatchingInfoPostUpgrade\_RS3** The total DataSourceMatchingInfoPostUpgrade objects targeting the next release of Windows on this device.
- **DecisionApplicationFile\_RS3** The total DecisionApplicationFile objects targeting the next release of Windows on this device.
- **DecisionDevicePnp\_RS3** The total DecisionDevicePnp objects targeting the next release of Windows on this device.
- **DecisionDriverPackage\_RS3** The total DecisionDriverPackage objects targeting the next release of Windows on this device.
- **DecisionMatchingInfoBlock\_RS3** The total DecisionMatchingInfoBlock objects targeting the next release of Windows on this device.
- **DecisionMatchingInfoPassive\_RS3** The total DataSourceMatchingInfoPassive objects targeting the next release of Windows on this device.
- **DecisionMatchingInfoPostUpgrade\_RS3** The total DecisionMatchingInfoPostUpgrade objects targeting the next release of Windows on this device.
- **DecisionMediaCenter\_RS3** The total DecisionMediaCenter objects targeting the next release of Windows on this device.
- **Wmdrm\_RS3** The total Wmdrm objects targeting the next release of Windows on this device.
- **DatasourceApplicationFile\_RS1** An ID for the system, calculated by hashing hardware identifiers.
- **DecisionApplicationFile\_RS1** An ID for the system, calculated by hashing hardware identifiers.
- **DataSourceMatchingInfoBlock\_RS1** The total DataSourceMatchingInfoBlock objects targeting Windows 10 version 1607 on this device.
- **DecisionMatchingInfoBlock\_RS1** The total DecisionMatchingInfoBlock objects targeting Windows 10 version 1607 present on this device.
- **DecisionMediaCenter\_RS1** The total DecisionMediaCenter objects targeting Windows 10 version 1607 present on this device.

#### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoBlockAdd**

This event sends blocking data about any compatibility blocking entries hit on the system that are not directly related to specific applications or devices, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.

#### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoBlockRemove**

This event indicates that the DataSourceMatchingInfoBlock object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoBlockStartSync**

This event indicates that a full set of DataSourceMatchingInfoBlockStAdd events have been sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPassiveAdd**

This event sends compatibility database information about non-blocking compatibility entries on the system that are not keyed by either applications or devices, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.

#### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPassiveRemove**

This event indicates that the DataSourceMatchingInfoPassive object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPassiveStartSync**

This event indicates that a new set of DataSourceMatchingInfoPassiveAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPostUpgradeAdd**

This event sends compatibility database information about entries requiring reinstallation after an upgrade on the system that are not keyed by either applications or devices, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.

#### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPostUpgradeRemove**

This event indicates that the DataSourceMatchingInfoPostUpgrade object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPostUpgradeStartSync**

This event indicates that a new set of DataSourceMatchingInfoPostUpgradeAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.DatasourceApplicationFileRemove**

This event indicates that the DatasourceApplicationFile object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.DatasourceApplicationFileStartSync**

This event indicates that a new set of DatasourceApplicationFileAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.DatasourceDevicePnpAdd**

This event sends compatibility data for a PNP device, to help keep Windows up-to-date.

The following fields are available:

- **ActiveNetworkConnection** Is the device an active network device?
- **AppraiserVersion** The version of the appraiser file generating the events.
- **IsBootCritical** Is the device boot critical?
- **SdbEntries** An array of fields indicating the SDB entries that apply to this device.
- **WuDriverCoverage** Is there a driver uplevel for this device according to Windows Update?
- **WuDriverUpdateId** The Windows Update ID of the applicable uplevel driver
- **WuPopulatedFromId** The expected up-level driver matching ID based on driver coverage from Windows Update

### **Microsoft.Windows.Appraiser.General.DatasourceDevicePnpRemove**

This event indicates that the DatasourceDevicePnp object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.DatasourceDevicePnpStartSync**

This event indicates that a new set of DatasourceDevicePnpAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.DatasourceDriverPackageAdd**

This event sends compatibility database data about driver packages to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.

### **Microsoft.Windows.Appraiser.General.DatasourceDriverPackageRemove**

This event indicates that the DatasourceDriverPackage object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.DatasourceDriverPackageStartSync**

This event indicates that a new set of DatasourceDriverPackageAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DatasourceSystemBiosAdd**

This event sends compatibility database information about the BIOS to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **SdbEntries** An array of fields indicating the SDB entries that apply to this BIOS.

#### **Microsoft.Windows.Appraiser.General.DatasourceSystemBiosRemove**

This event indicates that the DatasourceSystemBios object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DatasourceSystemBiosStartSync**

This event indicates that a new set of DatasourceSystemBiosAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionApplicationFileAdd**

This event sends compatibility decision data about a file to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **BlockAlreadyInbox** The uplevel runtime block on the file already existed on the current OS.
- **BlockingApplication** Are there any application issues that interfere with upgrade due to the file in question?
- **DisplayGenericMessage** Will be a generic message be shown for this file?
- **HardBlock** This file is blocked in the SDB.
- **HasUxBlockOverride** Does the file have a block that is overridden by a tag in the SDB?
- **MigApplication** Does the file have a MigXML from the SDB associated with it that applies to the current upgrade mode?
- **MigRemoval** Does the file have a MigXML from the SDB that will cause the app to be removed on upgrade?
- **NeedsDismissAction** Will the file cause an action that can be dismissed?
- **NeedsInstallPostUpgradeData** After upgrade, the file will have a post-upgrade notification to install a replacement for the app.
- **NeedsNotifyPostUpgradeData** Does the file have a notification that should be shown after upgrade?
- **NeedsReinstallPostUpgradeData** After upgrade, this file will have a post-upgrade notification to reinstall the app.
- **NeedsUninstallAction** The file must be uninstalled to complete the upgrade.
- **SdbBlockUpgrade** The file is tagged as blocking upgrade in the SDB,
- **SdbBlockUpgradeCanReinstall** The file is tagged as blocking upgrade in the SDB. It can be reinstalled after upgrade.
- **SdbBlockUpgradeUntilUpdate** The file is tagged as blocking upgrade in the SDB. If the app is updated, the upgrade can proceed.
- **SdbReinstallUpgrade** The file is tagged as needing to be reinstalled after upgrade in the SDB. It does not block upgrade.
- **SdbReinstallUpgradeWarn** The file is tagged as needing to be reinstalled after upgrade with a warning in the



SDB. It does not block upgrade.

- **SoftBlock** The file is softblocked in the SDB and has a warning.

#### **Microsoft.Windows.Appraiser.General.DecisionApplicationFileRemove**

This event indicates that the DecisionApplicationFile object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionApplicationFileStartSync**

This event indicates that a new set of DecisionApplicationFileAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionDevicePnpAdd**

This event sends compatibility decision data about a PNP device to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **AssociatedDriverIsBlocked** Is the driver associated with this PNP device blocked?
- **AssociatedDriverWillNotMigrate** Will the driver associated with this plug-and-play device migrate?
- **BlockAssociatedDriver** Should the driver associated with this PNP device be blocked?
- **BlockingDevice** Is this PNP device blocking upgrade?
- **BlockUpgradeIfDriverBlocked** Is the PNP device both boot critical and does not have a driver included with the OS?
- **BlockUpgradeIfDriverBlockedAndOnlyActiveNetwork** Is this PNP device the only active network device?
- **DisplayGenericMessage** Will a generic message be shown during Setup for this PNP device?
- **DriverAvailableInbox** Is a driver included with the operating system for this PNP device?
- **DriverAvailableOnline** Is there a driver for this PNP device on Windows Update?
- **DriverAvailableUplevel** Is there a driver on Windows Update or included with the operating system for this PNP device?
- **DriverBlockOverridden** Is there is a driver block on the device that has been overridden?
- **NeedsDismissAction** Will the user would need to dismiss a warning during Setup for this device?
- **NotRegressed** Does the device have a problem code on the source OS that is no better than the one it would have on the target OS?
- **SdbDeviceBlockUpgrade** Is there an SDB block on the PNP device that blocks upgrade?
- **SdbDriverBlockOverridden** Is there an SDB block on the PNP device that blocks upgrade, but that block was overridden?

#### **Microsoft.Windows.Appraiser.General.DecisionDevicePnpRemove**

This event indicates that the DecisionDevicePnp object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionDevicePnpStartSync**

This event indicates that the DecisionDevicePnp object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.DecisionDriverPackageAdd**

This event sends decision data about driver package compatibility to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **DriverBlockOverridden** Does the driver package have an SDB block that blocks it from migrating, but that block has been overridden?
- **DriverIsDeviceBlocked** Was the driver package was blocked because of a device block?
- **DriverIsDriverBlocked** Is the driver package blocked because of a driver block?
- **DriverShouldNotMigrate** Should the driver package be migrated during upgrade?
- **SdbDriverBlockOverridden** Does the driver package have an SDB block that blocks it from migrating, but that block has been overridden?

### **Microsoft.Windows.Appraiser.General.DecisionDriverPackageRemove**

This event indicates that the DecisionDriverPackage object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.DecisionDriverPackageStartSync**

This event indicates that a new set of DecisionDriverPackageAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoBlockAdd**

This event sends compatibility decision data about blocking entries on the system that are not keyed by either applications or devices, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **BlockingApplication** Are there are any application issues that interfere with upgrade due to matching info blocks?
- **DisplayGenericMessage** Will a generic message be shown for this block?
- **NeedsUninstallAction** Does the user need to take an action in setup due to a matching info block?
- **SdbBlockUpgrade** Is a matching info block blocking upgrade?
- **SdbBlockUpgradeCanReinstall** Is a matching info block blocking upgrade, but has the can reinstall tag?
- **SdbBlockUpgradeUntilUpdate** Is a matching info block blocking upgrade but has the until update tag?

### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoBlockRemove**

This event indicates that the DecisionMatchingInfoBlock object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoBlockStartSync**

This event indicates that a new set of DecisionMatchingInfoBlockAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPassiveAdd**

This event sends compatibility decision data about non-blocking entries on the system that are not keyed by either applications or devices, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **BlockingApplication** Are there any application issues that interfere with upgrade due to matching info blocks?
- **MigApplication** Is there a matching info block with a mig for the current mode of upgrade?

### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPassiveRemove**

This event Indicates that the DecisionMatchingInfoPassive object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPassiveStartSync**

This event indicates that a new set of DecisionMatchingInfoPassiveAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPostUpgradeAdd**

This event sends compatibility decision data about entries that require reinstall after upgrade. It's used to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **NeedsInstallPostUpgradeData** Will the file have a notification after upgrade to install a replacement for the app?
- **NeedsNotifyPostUpgradeData** Should a notification be shown for this file after upgrade?
- **NeedsReinstallPostUpgradeData** Will the file have a notification after upgrade to reinstall the app?
- **SdbReinstallUpgrade** The file is tagged as needing to be reinstalled after upgrade in the compatibility database (but is not blocking upgrade).

### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPostUpgradeRemove**

This event indicates that the DecisionMatchingInfoPostUpgrade object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.DecisionMediaCenterAdd**

This event sends decision data about the presence of Windows Media Center, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **BlockingApplication** Is there any application issues that interfere with upgrade due to Windows Media Center?
- **MediaCenterActivelyUsed** If Windows Media Center is supported on the edition, has it been run at least once and are the MediaCenterIndicators are true?
- **MediaCenterIndicators** Do any indicators imply that Windows Media Center is in active use?

- **MediaCenterInUse** Is Windows Media Center actively being used?
- **MediaCenterPaidOrActivelyUsed** Is Windows Media Center actively being used or is it running on a supported edition?
- **NeedsDismissAction** Are there any actions that can be dismissed coming from Windows Media Center?

#### **Microsoft.Windows.Appraiser.General.DecisionMediaCenterRemove**

This event indicates that the DecisionMediaCenter object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionMediaCenterStartSync**

This event indicates that a new set of DecisionMediaCenterAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionSystemBiosAdd**

This event sends compatibility decision data about the BIOS to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **Blocking** Is the device blocked from upgrade due to a BIOS block?
- **HasBiosBlock** Does the device have a BIOS block?

#### **Microsoft.Windows.Appraiser.General.DecisionSystemBiosRemove**

This event indicates that the DecisionSystemBios object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionSystemBiosStartSync**

This event indicates that a new set of DecisionSystemBiosAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.GatedRegChange**

This event sends data about the results of running a set of quick-blocking instructions, to help keep Windows up to date.

The following fields are available:

- **NewData** The data in the registry value after the scan completed.
- **OldData** The previous data in the registry value before the scan ran.
- **PCFP** An ID for the system calculated by hashing hardware identifiers.
- **RegKey** The registry key name for which a result is being sent.
- **RegValue** The registry value for which a result is being sent.
- **Time** The client time of the event.

#### **Microsoft.Windows.Appraiser.General.InventoryApplicationFileAdd**

This event represents the basic metadata about a file on the system. The file must be part of an app and either have a block in the compatibility database or are part of an anti-virus program.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **BinaryType** A binary type. Example: UNINITIALIZED, ZERO\_BYTE, DATA\_ONLY, DOS\_MODULE, NE16\_MODULE, PE32\_UNKNOWN, PE32\_I386, PE32\_ARM, PE64\_UNKNOWN, PE64\_AMD64, PE64\_ARM64, PE64\_IA64, PE32\_CLR\_32, PE32\_CLR\_IL, PE32\_CLR\_IL\_PREFER32, PE64\_CLR\_64
- **BinFileVersion** An attempt to clean up FileVersion at the client that tries to place the version into 4 octets.
- **BinProductVersion** An attempt to clean up ProductVersion at the client that tries to place the version into 4 octets.
- **BoeProgramId** If there is no entry in Add/Remove Programs, this is the ProgramID that is generated from the file metadata.
- **CompanyName** The company name of the vendor who developed this file.
- **FileId** A hash that uniquely identifies a file.
- **FileVersion** The File version field from the file metadata under Properties -> Details.
- **LinkDate** The date and time that this file was linked on.
- **LowerCaseLongPath** The full file path to the file that was inventoried on the device.
- **Name** The name of the file that was inventoried.
- **ProductName** The Product name field from the file metadata under Properties -> Details.
- **ProductVersion** The Product version field from the file metadata under Properties -> Details.
- **ProgramId** A hash of the Name, Version, Publisher, and Language of an application used to identify it.
- **Size** The size of the file (in hexadecimal bytes).

#### **Microsoft.Windows.Appraiser.General.InventoryApplicationFileRemove**

This event indicates that the InventoryApplicationFile object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.InventoryApplicationFileStartSync**

This event indicates that a new set of InventoryApplicationFileAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.InventoryLanguagePackAdd**

This event sends data about the number of language packs installed on the system, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **HasLanguagePack** Does this device have 2 or more language packs?
- **LanguagePackCount** How many language packs are installed?

#### **Microsoft.Windows.Appraiser.General.InventoryLanguagePackRemove**

This event indicates that the InventoryLanguagePack object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.InventoryLanguagePackStartSync**

This event indicates that a new set of InventoryLanguagePackAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.InventoryMediaCenterAdd**

This event sends true/false data about decision points used to understand whether Windows Media Center is used on the system, to help keep Windows up to date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **EverLaunched** Has Windows Media Center ever been launched?
- **HasConfiguredTv** Has the user configured a TV tuner through Windows Media Center?
- **HasExtendedUserAccounts** Are any Windows Media Center Extender user accounts configured?
- **HasWatchedFolders** Are any folders configured for Windows Media Center to watch?
- **IsDefaultLauncher** Is Windows Media Center the default app for opening music or video files?
- **IsPaid** Is the user running a Windows Media Center edition that implies they paid for Windows Media Center?
- **IsSupported** Does the running OS support Windows Media Center?

#### **Microsoft.Windows.Appraiser.General.InventoryMediaCenterRemove**

This event indicates that the InventoryMediaCenter object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.InventoryMediaCenterStartSync**

This event indicates that a new set of InventoryMediaCenterAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.InventorySystemBiosAdd**

This event sends basic metadata about the BIOS to determine whether it has a compatibility block.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **BiosDate** The release date of the BIOS in UTC format.
- **BiosName** The name field from Win32\_BIOS.
- **Manufacturer** The manufacturer field from Win32\_ComputerSystem.
- **Model** The model field from Win32\_ComputerSystem.

#### **Microsoft.Windows.Appraiser.General.InventorySystemBiosRemove**

This event indicates that the InventorySystemBios object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.InventorySystemBiosStartSync**

This event indicates that a new set of InventorySystemBiosAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.InventoryUplevelDriverPackageRemove**

This event indicates that the InventoryUplevelDriverPackage object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.InventoryUplevelDriverPackageStartSync**

This event indicates that a new set of InventoryUplevelDriverPackageAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.RunContext**

This event indicates what should be expected in the data payload.

The following fields are available:

- **AppraiserBranch** The source branch in which the currently running version of Appraiser was built.
- **AppraiserProcess** The name of the process that launched Appraiser.
- **AppraiserVersion** The version of the Appraiser file generating the events.
- **Context** Indicates what mode Appraiser is running in. Example: Setup or Telemetry.
- **PCFP** An ID for the system calculated by hashing hardware identifiers.
- **Time** The client time of the event.

### **Microsoft.Windows.Appraiser.General.SystemMemoryAdd**

This event sends data on the amount of memory on the system and whether it meets requirements, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **Blocking** Is the device from upgrade due to memory restrictions?
- **MemoryRequirementViolated** Was a memory requirement violated?
- **pageFile** The current committed memory limit for the system or the current process, whichever is smaller (in bytes).
- **ram** The amount of memory on the device.
- **ramKB** The amount of memory (in KB).
- **virtual** The size of the user-mode portion of the virtual address space of the calling process (in bytes).
- **virtualKB** The amount of virtual memory (in KB).

### **Microsoft.Windows.Appraiser.General.SystemMemoryRemove**

This event that the SystemMemory object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.SystemMemoryStartSync**

This event indicates that a new set of SystemMemoryAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.SystemProcessorCompareExchangeAdd**

This event sends data indicating whether the system supports the CompareExchange128 CPU requirement, to help keep Windows up to date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **Blocking** Is the upgrade blocked due to the processor?
- **CompareExchange128Support** Does the CPU support CompareExchange128?

#### **Microsoft.Windows.Appraiser.General.SystemProcessorCompareExchangeRemove**

This event indicates that the SystemProcessorCompareExchange object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorCompareExchangeStartSync**

This event indicates that a new set of SystemProcessorCompareExchangeAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorLahfSahfAdd**

This event sends data indicating whether the system supports the LahfSahf CPU requirement, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **Blocking** Is the upgrade blocked due to the processor?
- **LahfSahfSupport** Does the CPU support LAHF/SAHF?

#### **Microsoft.Windows.Appraiser.General.SystemProcessorLahfSahfRemove**

This event indicates that the SystemProcessorLahfSahf object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorLahfSahfStartSync**

This event indicates that a new set of SystemProcessorLahfSahfAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorNxAdd**

This event sends data indicating whether the system supports the NX CPU requirement, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **Blocking** Is the upgrade blocked due to the processor?
- **NXDriverResult** The result of the driver used to do a non-deterministic check for NX support.
- **NXProcessorSupport** Does the processor support NX?

#### **Microsoft.Windows.Appraiser.General.SystemProcessorNxRemove**



This event indicates that the SystemProcessorNx object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorNxStartSync**

This event indicates that a new set of SystemProcessorNxAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorPrefetchWAdd**

This event sends data indicating whether the system supports the PrefetchW CPU requirement, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **Blocking** Is the upgrade blocked due to the processor?
- **PrefetchWSupport** Does the processor support PrefetchW?

#### **Microsoft.Windows.Appraiser.General.SystemProcessorPrefetchWRemove**

This event indicates that the SystemProcessorPrefetchW object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorPrefetchWStartSync**

This event indicates that a new set of SystemProcessorPrefetchWAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorSse2Add**

This event sends data indicating whether the system supports the SSE2 CPU requirement, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **Blocking** Is the upgrade blocked due to the processor?
- **SSE2ProcessorSupport** Does the processor support SSE2?

#### **Microsoft.Windows.Appraiser.General.SystemProcessorSse2Remove**

This event indicates that the SystemProcessorSse2 object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorSse2StartSync**

This event indicates that a new set of SystemProcessorSse2Add events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.SystemTouchAdd**

This event sends data indicating whether the system supports touch, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **IntegratedTouchDigitizerPresent** Is there an integrated touch digitizer?
- **MaximumTouches** The maximum number of touch points supported by the device hardware.

### **Microsoft.Windows.Appraiser.General.SystemTouchRemove**

This event indicates that the SystemTouch object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.SystemTouchStartSync**

This event indicates that a new set of SystemTouchAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.SystemWimAdd**

This event sends data indicating whether the operating system is running from a compressed WIM file, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **IsWimBoot** Is the current operating system running from a compressed WIM file?
- **RegistryWimBootValue** The raw value from the registry that is used to indicate if the device is running from a WIM.

### **Microsoft.Windows.Appraiser.General.SystemWimRemove**

This event indicates that the SystemWim object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.SystemWimStartSync**

This event indicates that a new set of SystemWimAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.SystemWindowsActivationStatusAdd**

This event sends data indicating whether the current operating system is activated, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **WindowsIsLicensedApiValue** The result from the API that's used to indicate if operating system is activated.
- **WindowsNotActivatedDecision** Is the current operating system activated?

### **Microsoft.Windows.Appraiser.General.SystemWindowsActivationStatusRemove**

This event indicates that the SystemWindowsActivationStatus object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemWindowsActivationStatusStartSync**

This event indicates that a new set of SystemWindowsActivationStatusAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemWlanRemove**

This event indicates that the SystemWlan object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemWlanStartSync**

This event indicates that a new set of SystemWlanAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.TelemetryRunHealth**

A summary event indicating the parameters and result of a telemetry run. This allows the rest of the data sent over the course of the run to be properly contextualized and understood, which is then used to keep Windows up-to-date.

The following fields are available:

- **AppraiserBranch** The source branch in which the version of Appraiser that is running was built.
- **AppraiserDataVersion** The version of the data files being used by the Appraiser telemetry run.
- **AppraiserProcess** The name of the process that launched Appraiser.
- **AppraiserVersion** The file version (major, minor and build) of the Appraiser DLL, concatenated without dots.
- **AuxFinal** Obsolete, always set to false
- **AuxInitial** Obsolete, indicates if Appraiser is writing data files to be read by the Get Windows 10 app.
- **DeadlineDate** A timestamp representing the deadline date, which is the time until which appraiser will wait to do a full scan.
- **EnterpriseRun** Indicates if the telemetry run is an enterprise run, which means appraiser was run from the command line with an extra enterprise parameter.
- **FullSync** Indicates if Appraiser is performing a full sync, which means that full set of events representing the state of the machine are sent. Otherwise, only the changes from the previous run are sent.
- **InventoryFullSync** Indicates if inventory is performing a full sync, which means that the full set of events representing the inventory of machine are sent.
- **PCFP** An ID for the system calculated by hashing hardware identifiers.
- **PerfBackoff** Indicates if the run was invoked with logic to stop running when a user is present. Helps to understand why a run may have a longer elapsed time than normal.
- **PerfBackoffInsurance** Indicates if appraiser is running without performance backoff because it has run with perf backoff and failed to complete several times in a row.
- **RunAppraiser** Indicates if Appraiser was set to run at all. If this is false, it is understood that data events will not be received from this device.
- **RunDate** The date that the telemetry run was stated, expressed as a filetime.

- **RunGeneralTel** Indicates if the generaltel.dll component was run. Generaltel collects additional telemetry on an infrequent schedule and only from machines at telemetry levels higher than Basic.
- **RunOnline** Indicates if appraiser was able to connect to Windows Update and therefore is making decisions using up-to-date driver coverage information.
- **RunResult** The hresult of the Appraiser telemetry run.
- **SendingUtc** Indicates if the Appraiser client is sending events during the current telemetry run.
- **StoreHandlesNotNull** Obsolete, always set to false
- **TelemetrySent** Indicates if telemetry was successfully sent.
- **ThrottlingUtc** Indicates if the Appraiser client is throttling its output of CUET events to avoid being disabled. This increases runtime but also telemetry reliability.
- **Time** The client time of the event.
- **VerboseMode** Indicates if appraiser ran in Verbose mode, which is a test-only mode with extra logging.
- **WhyFullSyncWithoutTablePrefix** Indicates the reason or reasons that a full sync was generated.

#### **Microsoft.Windows.Appraiser.General.WmdrmAdd**

This event sends data about the usage of older digital rights management on the system, to help keep Windows up to date. This data does not indicate the details of the media using the digital rights management, only whether any such files exist. Collecting this data was critical to ensuring the correct mitigation for customers, and should be able to be removed once all mitigations are in place.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **BlockingApplication** Same as NeedsDismissAction
- **NeedsDismissAction** Indicates if a dismissible message is needed to warn the user about a potential loss of data due to DRM deprecation.
- **WmdrmApiResult** Raw value of the API used to gather DRM state.
- **WmdrmCdRipped** Indicates if the system has any files encrypted with personal DRM, which was used for ripped CDs.
- **WmdrmIndicators** WmdrmCdRipped OR WmdrmPurchased
- **WmdrmInUse** WmdrmIndicators AND dismissible block in setup was not dismissed.
- **WmdrmNonPermanent** Indicates if the system has any files with non-permanent licenses.
- **WmdrmPurchased** Indicates if the system has any files with permanent licenses.

#### **Microsoft.Windows.Appraiser.General.WmdrmRemove**

This event indicates that the Wmdrm object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.WmdrmStartSync**

This event indicates that a new set of WmdrmAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

## Census events

### **Census.App**

This event sends version data about the Apps running on this device, to help keep Windows up to date.

The following fields are available:

- **CensusVersion** The version of Census that generated the current data for this device.
- **IEVersion** Retrieves which version of Internet Explorer is running on this device.

### Census.Battery

This event sends type and capacity data about the battery on the device, as well as the number of connected standby devices in use, type to help keep Windows up to date.

The following fields are available:

- **InternalBatteryCapabilities** Represents information about what the battery is capable of doing.
- **InternalBatteryCapacityCurrent** Represents the battery's current fully charged capacity in mWh (or relative). Compare this value to `DesignedCapacity` to estimate the battery's wear.
- **InternalBatteryCapacityDesign** Represents the theoretical capacity of the battery when new, in mWh.
- **InternalBatteryNumberOfCharges** Provides the number of battery charges. This is used when creating new products and validating that existing products meets targeted functionality performance.
- **IsAlwaysOnAlwaysConnectedCapable** Represents whether the battery enables the device to be `AlwaysOnAlwaysConnected`. Boolean value.

### Census.Camera

This event sends data about the resolution of cameras on the device, to help keep Windows up to date.

The following fields are available:

- **FrontFacingCameraResolution** Represents the resolution of the front facing camera in megapixels. If a front facing camera does not exist, then the value is 0.
- **RearFacingCameraResolution** Represents the resolution of the rear facing camera in megapixels. If a rear facing camera does not exist, then the value is 0.

### Census.Enterprise

This event sends data about Azure presence, type, and cloud domain use in order to provide an understanding of the use and integration of devices in an enterprise, cloud, and server environment.

The following fields are available:

- **AzureOSIDPresent** Represents the field used to identify an Azure machine.
- **AzureVMType** Represents whether the instance is Azure VM PAAS, Azure VM IAAS or any other VMs.
- **CDJType** Represents the type of cloud domain joined for the machine.
- **CommercialId** Represents the GUID for the commercial entity which the device is a member of. Will be used to reflect insights back to customers.
- **ContainerType** The type of container, such as process or virtual machine hosted.
- **EnrollmentType** Defines the type of MDM enrollment on the device.
- **HashedDomain** The hashed representation of the user domain used for login.
- **IsCloudDomainJoined** Is this device joined to an Azure Active Directory (AAD) tenant? true/false
- **IsDERequirementMet** Represents if the device can do device encryption.
- **IsDeviceProtected** Represents if Device protected by BitLocker/Device Encryption
- **IsDomainJoined** Indicates whether a machine is joined to a domain.
- **IsEDPEnabled** Represents if Enterprise data protected on the device.
- **IsMDMEnrolled** Whether the device has been MDM Enrolled or not.
- **MPNId** Returns the Partner ID/MPN ID from Regkey.  
HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\DeployID
- **SCCMClientId** This ID correlate systems that send data to Compat Analytics (OMS) and other OMS based systems with systems in an Enterprise SCCM environment.
- **ServerFeatures** Represents the features installed on a Windows Server. This can be used by developers and

administrators who need to automate the process of determining the features installed on a set of server computers.

- **SystemCenterID** The SCCM ID is an anonymized one-way hash of the Active Directory Organization identifier

### Census.Firmware

This event sends data about the BIOS and startup embedded in the device, to help keep Windows up to date.

The following fields are available:

- **FirmwareManufacturer** Represents the manufacturer of the device's firmware (BIOS).
- **FirmwareReleaseDate** Represents the date the current firmware was released.
- **FirmwareType** Represents the firmware type. The various types can be unknown, BIOS, UEFI.
- **FirmwareVersion** Represents the version of the current firmware.

### Census.Flighting

This event sends Windows Insider data from customers participating in improvement testing and feedback programs, to help keep Windows up-to-date.

The following fields are available:

- **DeviceSampleRate** The telemetry sample rate assigned to the device.
- **EnablePreviewBuilds** Used to enable Windows Insider builds on a device.
- **FlightIds** A list of the different Windows Insider builds on this device.
- **FlightingBranchName** The name of the Windows Insider branch currently used by the device.
- **IsFlightsDisabled** Represents if the device is participating in the Windows Insider program.
- **MSA\_Accounts** Represents a list of hashed IDs of the Microsoft Accounts that are flighting (pre-release builds) on this device.
- **SSRK** Retrieves the mobile targeting settings.

### Census.Hardware

This event sends data about the device, including hardware type, OEM brand, model line, model, telemetry level setting, and TPM support, to help keep Windows up-to-date.

The following fields are available:

- **ActiveMicCount** The number of active microphones attached to the device.
- **ChassisType** Represents the type of device chassis, such as desktop or low profile desktop. The possible values can range between 1 - 36.
- **ComputerHardwareID** Identifies a device class that is represented by a hash of different SMBIOS fields.
- **D3DMaxFeatureLevel** Supported Direct3D version.
- **DeviceForm** Indicates the form as per the device classification.
- **DeviceName** The device name that is set by the user.
- **DigitizerSupport** Is a digitizer supported?
- **DUID** The device unique ID.
- **Gyroscope** Indicates whether the device has a gyroscope (a mechanical component that measures and maintains orientation).
- **InventoryId** The device ID used for compatibility testing.
- **Magnetometer** Indicates whether the device has a magnetometer (a mechanical component that works like a compass).
- **NFCProximity** Indicates whether the device supports NFC (a set of communication protocols that helps establish communication when applicable devices are brought close together).
- **OEMDigitalMarkerFileName** The name of the file placed in the \Windows\system32\drivers directory that specifies the OEM and model name of the device.

- **OEMManufacturerName** The device manufacturer name. The OEMName for an inactive device is not reprocessed even if the clean OEM name is changed at a later date.
- **OEMModelBaseBoard** The baseboard model used by the OEM.
- **OEMModelBaseBoardVersion** Differentiates between developer and retail devices.
- **OEMModelName** The device model name.
- **OEMModelNumber** The device model number.
- **OEMModelSKU** The device edition that is defined by the manufacturer.
- **OEMModelSystemFamily** The system family set on the device by an OEM.
- **OEMModelSystemVersion** The system model version set on the device by the OEM.
- **OEMOptionalIdentifier** A Microsoft assigned value that represents a specific OEM subsidiary.
- **OEMSerialNumber** The serial number of the device that is set by the manufacturer.
- **PhoneManufacturer** The friendly name of the phone manufacturer.
- **PowerPlatformRole** The OEM preferred power management profile. It's used to help to identify the basic form factor of the device.
- **SoCName** The firmware manufacturer of the device.
- **StudyID** Used to identify retail and non-retail device.
- **TelemetryLevel** The telemetry level the user has opted into, such as Basic or Enhanced.
- **TelemetryLevelLimitEnhanced** The telemetry level for Windows Analytics-based solutions.
- **TelemetrySettingAuthority** Determines who set the telemetry level, such as GP, MDM, or the user.
- **TPMVersion** The supported Trusted Platform Module (TPM) on the device. If no TPM is present, the value is 0.
- **VoiceSupported** Does the device have a cellular radio capable of making voice calls?
- **DeviceColor** Indicates a color of the device.

### Census.Memory

This event sends data about the memory on the device, including ROM and RAM, to help keep Windows up to date.

The following fields are available:

- **TotalPhysicalRAM** Represents the physical memory (in MB).
- **TotalVisibleMemory** Represents the memory that is not reserved by the system.

### Census.Network

This event sends data about the mobile and cellular network used by the device (mobile service provider, network, device ID, and service cost factors), to help keep Windows up to date.

The following fields are available:

- **IMEI0** Represents the International Mobile Station Equipment Identity. This number is usually unique and used by the mobile operator to distinguish different phone hardware. Microsoft does not have access to mobile operator billing data so collecting this data does not expose or identify the user. The two fields represent phone with dual sim coverage.
- **IMEI1** Represents the International Mobile Station Equipment Identity. This number is usually unique and used by the mobile operator to distinguish different phone hardware. Microsoft does not have access to mobile operator billing data so collecting this data does not expose or identify the user. The two fields represent phone with dual sim coverage.
- **MCC0** Represents the Mobile Country Code (MCC). It used with the Mobile Network Code (MNC) to uniquely identify a mobile network operator. The two fields represent phone with dual sim coverage.
- **MCC1** Represents the Mobile Country Code (MCC). It used with the Mobile Network Code (MNC) to uniquely identify a mobile network operator. The two fields represent phone with dual sim coverage.
- **MEID** Represents the Mobile Equipment Identity (MEID). MEID is a worldwide unique phone ID assigned to

CDMA phones. MEID replaces electronic serial number (ESN), and is equivalent to IMEI for GSM and WCDMA phones. Microsoft does not have access to mobile operator billing data so collecting this data does not expose or identify the user.

- **MNC0** Retrieves the Mobile Network Code (MNC). It used with the Mobile Country Code (MCC) to uniquely identify a mobile network operator. The two fields represent phone with dual sim coverage.
- **MNC1** Retrieves the Mobile Network Code (MNC). It used with the Mobile Country Code (MCC) to uniquely identify a mobile network operator. The two fields represent phone with dual sim coverage.
- **MobileOperatorBilling** Represents the telephone company that provides services for mobile phone users.
- **MobileOperatorCommercialized** Represents which reseller and geography the phone is commercialized for. This is the set of values on the phone for who and where it was intended to be used. For example, the commercialized mobile operator code AT&T in the US would be ATT-US.
- **MobileOperatorNetwork0** Represents the operator of the current mobile network that the device is used on. (AT&T, T-Mobile, Vodafone). The two fields represent phone with dual sim coverage.
- **MobileOperatorNetwork1** Represents the operator of the current mobile network that the device is used on. (AT&T, T-Mobile, Vodafone). The two fields represent phone with dual sim coverage.
- **NetworkAdapterGUID** The GUID of the primary network adapter.
- **NetworkCost** Represents the network cost associated with a connection.
- **SPN0** Retrieves the Service Provider Name (SPN). For example, these might be AT&T, Sprint, T-Mobile, or Verizon. The two fields represent phone with dual sim coverage.
- **SPN1** Retrieves the Service Provider Name (SPN). For example, these might be AT&T, Sprint, T-Mobile, or Verizon. The two fields represent phone with dual sim coverage.

## Census.OS

This event sends data about the operating system such as the version, locale, update service configuration, when and how it was originally installed, and whether it is a virtual device, to help keep Windows up to date.

The following fields are available:

- **ActivationChannel** Retrieves the retail license key or Volume license key for a machine.
- **AssignedAccessStatus** Kiosk configuration mode.
- **CompactOS** Indicates if the Compact OS feature from Win10 is enabled.
- **DeveloperUnlockStatus** Represents if a device has been developer unlocked by the user or Group Policy.
- **DeviceTimeZone** The time zone that is set on the device. Example: Pacific Standard Time
- **GenuineState** Retrieves the ID Value specifying the OS Genuine check.
- **InstallationType** Retrieves the type of OS installation. (Clean, Upgrade, Reset, Refresh, Update).
- **InstallLanguage** The first language installed on the user machine.
- **IsDeviceRetailDemo** Retrieves if the device is running in demo mode.
- **IsEduData** Returns Boolean if the education data policy is enabled.
- **IsPortableOperatingSystem** Retrieves whether OS is running Windows-To-Go
- **IsSecureBootEnabled** Retrieves whether Boot chain is signed under UEFI.
- **LanguagePacks** The list of language packages installed on the device.
- **LicenseStateReason** Retrieves why (or how) a system is licensed or unlicensed. The HRESULT may indicate an error code that indicates a key blocked error, or it may indicate that we are running an OS License granted by the MS store.
- **OA3xOriginalProductKey** Retrieves the License key stamped by the OEM to the machine.
- **OSEdition** Retrieves the version of the current OS.
- **OSInstallType** Retrieves a numeric description of what install was used on the device i.e. clean, upgrade, refresh, reset, etc
- **OSOBEDateTime** Retrieves Out of Box Experience (OOBE) Date in Coordinated Universal Time (UTC).
- **OSSKU** Retrieves the Friendly Name of OS Edition.



- **OSSubscriptionStatus** Represents the existing status for enterprise subscription feature for PRO machines.
- **OSSubscriptionTypeId** Returns boolean for enterprise subscription feature for selected PRO machines.
- **OSTimeZoneBiasInMins** Retrieves the time zone set on machine.
- **OSUILocale** Retrieves the locale of the UI that is currently used by the OS.
- **ProductActivationResult** Returns Boolean if the OS Activation was successful.
- **ProductActivationTime** Returns the OS Activation time for tracking piracy issues.
- **ProductKeyID2** Retrieves the License key if the machine is updated with a new license key.
- **RACw7Id** Retrieves the Microsoft Reliability Analysis Component (RAC) Win7 Identifier. RAC is used to monitor and analyze system usage and reliability.
- **ServiceMachineIP** Retrieves the IP address of the KMS host used for anti-piracy.
- **ServiceMachinePort** Retrieves the port of the KMS host used for anti-piracy.
- **ServiceProductKeyID** Retrieves the License key of the KMS
- **SharedPCMode** Returns Boolean for education devices used as shared cart
- **Signature** Retrieves if it is a signature machine sold by Microsoft store.
- **SLICStatus** Whether a SLIC table exists on the device.
- **SLICVersion** Returns OS type/version from SLIC table.

### Census.Processor

This event sends data about the processor (architecture, speed, number of cores, manufacturer, and model number), to help keep Windows up to date.

The following fields are available:

- **KvaShadow** Microcode info of the processor.
- **MMSettingOverride** Microcode setting of the processor.
- **MMSettingOverrideMask** Microcode setting override of the processor.
- **ProcessorArchitecture** Processor architecture of the installed operating system.
- **ProcessorClockSpeed** Clock speed of the processor in MHz.
- **ProcessorCores** Number of logical cores in the processor.
- **ProcessorIdentifier** Processor Identifier of a manufacturer.
- **ProcessorManufacturer** Name of the processor manufacturer.
- **ProcessorModel** Name of the processor model.
- **ProcessorPhysicalCores** Number of physical cores in the processor.
- **ProcessorUpdateRevision** Microcode revision.
- **ProcessorUpdateStatus** The status of the microcode update.
- **SocketCount** Count of CPU sockets.
- **SpeculationControl** If the system has enabled protections needed to validate the speculation control vulnerability.

### Census.Security

This event provides information on about security settings used to help keep Windows up-to-date and secure.

The following fields are available:

- **AvailableSecurityProperties** This field helps to enumerate and report state on the relevant security properties for Device Guard
- **CGRunning** Credential Guard isolates and hardens key system and user secrets against compromise, helping to minimize the impact and breadth of a Pass the Hash style attack in the event that malicious code is already running via a local or network based vector. This field tells if Credential Guard is running.
- **DGState** This field summarizes Device Guard state
- **HVCIRunning** Hypervisor Code Integrity (HVCI) enables Device Guard to help protect kernel mode processes

and drivers from vulnerability exploits and zero days. HVCI uses the processor's functionality to force all software running in kernel mode to safely allocate memory. This field tells if HVCI is running

- **RequiredSecurityProperties** This field describes the required security properties to enable virtualization-based security
- **SecureBootCapable** Systems that support Secure Boot can have the feature turned off via BIOS. This field tells if the system is capable of running Secure Boot, regardless of the BIOS setting.
- **VBSState** Virtualization-based security (VBS) uses the hypervisor to help protect the kernel and other parts of the operating system. Credential Guard and Hypervisor Code Integrity (HVCI) both depend on VBS to isolate/protect secrets, and kernel-mode code integrity validation. VBS has a tri-state that can be Disabled, Enabled, or Running.

### Census.Speech

This event is used to gather basic speech settings on the device.

The following fields are available:

- **AboveLockEnabled** Cortana setting that represents if Cortana can be invoked when the device is locked.
- **GPAllowInputPersonalization** Indicates if a Group Policy setting has enabled speech functionalities.
- **HolographicSpeechInputDisabled** Holographic setting that represents if the attached HMD devices have speech functionality disabled by the user.
- **HolographicSpeechInputDisabledRemote** Indicates if a remote policy has disabled speech functionalities for the HMD devices.
- **KWSEnabled** Cortana setting that represents if a user has enabled the "Hey Cortana" keyword spotter (KWS).
- **MDMAllowInputPersonalization** Indicates if an MDM policy has enabled speech functionalities.
- **RemotelyManaged** Indicates if the device is being controlled by a remote administrator (MDM or Group Policy) in the context of speech functionalities.
- **SpeakerIdEnabled** Cortana setting that represents if keyword detection has been trained to try to respond to a single user's voice.
- **SpeechServicesEnabled** Windows setting that represents whether a user is opted-in for speech services on the device.

### Census.Storage

This event sends data about the total capacity of the system volume and primary disk, to help keep Windows up to date.

The following fields are available:

- **PrimaryDiskTotalCapacity** Retrieves the amount of disk space on the primary disk of the device in MB.
- **PrimaryDiskType** Retrieves an enumerator value of type STORAGE\_BUS\_TYPE that indicates the type of bus to which the device is connected. This should be used to interpret the raw device properties at the end of this structure (if any).
- **SystemVolumeTotalCapacity** Retrieves the size of the partition that the System volume is installed on in MB.

### Census.UserDisplay

This event sends data about the logical/physical display size, resolution and number of internal/external displays, and VRAM on the system, to help keep Windows up to date.

The following fields are available:

- **InternalPrimaryDisplayLogicalDPIX** Retrieves the logical DPI in the x-direction of the internal display.
- **InternalPrimaryDisplayLogicalDPIY** Retrieves the logical DPI in the y-direction of the internal display.
- **InternalPrimaryDisplayPhysicalDPIX** Retrieves the physical DPI in the x-direction of the internal display.
- **InternalPrimaryDisplayPhysicalDPIY** Retrieves the physical DPI in the y-direction of the internal display.

- **InternalPrimaryDisplayResolutionHorizontal** Retrieves the number of pixels in the horizontal direction of the internal display.
- **InternalPrimaryDisplayResolutionVertical** Retrieves the number of pixels in the vertical direction of the internal display.
- **InternalPrimaryDisplaySizePhysicalH** Retrieves the physical horizontal length of the display in mm. Used for calculating the diagonal length in inches .
- **InternalPrimaryDisplaySizePhysicalY** Retrieves the physical vertical length of the display in mm. Used for calculating the diagonal length in inches
- **NumberOfExternalDisplays** Retrieves the number of external displays connected to the machine
- **NumberOfInternalDisplays** Retrieves the number of internal displays in a machine.
- **VRAMDedicated** Retrieves the video RAM in MB.
- **VRAMDedicatedSystem** Retrieves the amount of memory on the dedicated video card.
- **VRAMSharedSystem** Retrieves the amount of RAM memory that the video card can use.

### Census.UserNLS

This event sends data about the default app language, input, and display language preferences set by the user, to help keep Windows up to date.

The following fields are available:

- **DefaultAppLanguage** The current user Default App Language.
- **DisplayLanguage** The current user preferred Windows Display Language.
- **HomeLocation** The current user location, which is populated using GetUserGeold() function.
- **KeyboardInputLanguages** The Keyboard input languages installed on the device.
- **SpeechInputLanguages** The Speech Input languages installed on the device.

### Census.Userdefault

This event sends data about the current user's default preferences for browser and several of the most popular extensions and protocols, to help keep Windows up to date.

The following fields are available:

- **DefaultApp** The current uer's default program selected for the following extension or protocol:  
.html,.htm,.jpg,.jpeg,.png,.mp3,.mp4, .mov,.pdf
- **DefaultBrowserProgId** The ProgramId of the current user's default browser

### Census.VM

This event sends data indicating whether virtualization is enabled on the device, and its various characteristics, to help keep Windows up to date.

The following fields are available:

- **CloudService** Indicates which cloud service, if any, that this virtual machine is running within.
- **HyperVisor** Retrieves whether the current OS is running on top of a Hypervisor.
- **IOMMUPresent** Represents if an input/output memory management unit (IOMMU) is present.
- **IsVDI** Is the device using Virtual Desktop Infrastructure?
- **IsVirtualDevice** Retrieves that when the Hypervisor is Microsoft's Hyper-V Hypervisor or other Hv#1 Hypervisor, this field will be set to FALSE for the Hyper-V host OS and TRUE for any guest OS's. This field should not be relied upon for non-Hv#1 Hypervisors.
- **SLATSupported** Represents whether Second Level Address Translation (SLAT) is supported by the hardware.
- **VirtualizationFirmwareEnabled** Represents whether virtualization is enabled in the firmware.

### Census.WU

This event sends data about the Windows update server and other App store policies, to help keep Windows up to date.

The following fields are available:

- **AppraiserGatedStatus** Indicates whether a device has been gated for upgrading.
- **AppStoreAutoUpdate** Retrieves the Appstore settings for auto upgrade. (Enable/Disabled).
- **AppStoreAutoUpdateMDM** Retrieves the App Auto Update value for MDM: 0 - Disallowed. 1 - Allowed. 2 - Not configured. Default: [2] Not configured
- **AppStoreAutoUpdatePolicy** Retrieves the Microsoft Store App Auto Update group policy setting
- **DelayUpgrade** Retrieves the Windows upgrade flag for delaying upgrades.
- **OSAssessmentFeatureOutOfDate** How many days has it been since a the last feature update was released but the device did not install it?
- **OSAssessmentForFeatureUpdate** Is the device is on the latest feature update?
- **OSAssessmentForQualityUpdate** Is the device on the latest quality update?
- **OSAssessmentForSecurityUpdate** Is the device on the latest security update?
- **OSAssessmentQualityOutOfDate** How many days has it been since a the last quality update was released but the device did not install it?
- **OSAssessmentReleaseInfoTime** The freshness of release information used to perform an assessment.
- **OSRollbackCount** The number of times feature updates have rolled back on the device.
- **OSRolledBack** A flag that represents when a feature update has rolled back during setup.
- **OSUninstalled** A flag that represents when a feature update is uninstalled on a device .
- **OSWUAutoUpdateOptions** Retrieves the auto update settings on the device.
- **UninstallActive** A flag that represents when a device has uninstalled a previous upgrade recently.
- **UpdateServiceURLConfigured** Retrieves if the device is managed by Windows Server Update Services (WSUS).
- **WUDeferUpdatePeriod** Retrieves if deferral is set for Updates
- **WUDeferUpgradePeriod** Retrieves if deferral is set for Upgrades
- **WUDODownloadMode** Retrieves whether DO is turned on and how to acquire/distribute updates Delivery Optimization (DO) allows users to deploy previously downloaded WU updates to other devices on the same network.
- **WUMachineId** Retrieves the Windows Update (WU) Machine Identifier.
- **WUPauseState** Retrieves WU setting to determine if updates are paused
- **WUServer** Retrieves the HTTP(S) URL of the WSUS server that is used by Automatic Updates and API callers (by default).

### Census.Xbox

This event sends data about the Xbox Console, such as Serial Number and DeviceId, to help keep Windows up to date.

The following fields are available:

- **XboxConsolePreferredLanguage** Retrieves the preferred language selected by the user on Xbox console.
- **XboxConsoleSerialNumber** Retrieves the serial number of the Xbox console.
- **XboxLiveDeviceId** Retrieves the unique device id of the console.
- **XboxLiveSandboxId** Retrieves the developer sandbox id if the device is internal to MS.

## Deployment events

### DeploymentTelemetry.Deployment\_End

Event to indicate that a Deployment 360 API has completed.

The following fields are available:

- **ClientId** Client ID of user utilizing the D360 API
- **ErrorCode** Error code of action
- **FlightId** Flight being used
- **Mode** Phase in upgrade
- **RelatedCV** CV of any other related events
- **Result** End result of action

#### **DeploymentTelemetry.Deployment\_Initialize**

Event to indicate that the Deployment 360 APIs have been initialized for use.

The following fields are available:

- **ClientId** Client ID of user utilizing the D360 API
- **ErrorCode** Error code of action
- **FlightId** Flight being used
- **RelatedCV** CV of any other related events
- **Result** Phase Setup is in

#### **DeploymentTelemetry.Deployment\_SetupBoxLaunch**

Event to indicate that the Deployment 360 APIs have launched Setup Box.

The following fields are available:

- **ClientId** Client ID of user utilizing the D360 API
- **FlightId** Flight being used
- **Quiet** Whether Setup will run in quiet mode or in full
- **RelatedCV** CV of any other related events
- **SetupMode** Phase Setup is in

#### **DeploymentTelemetry.Deployment\_SetupBoxResult**

Event to indicate that the Deployment 360 APIs have received a return from Setup Box.

The following fields are available:

- **ClientId** Client ID of user utilizing the D360 API
- **ErrorCode** Error code of action
- **FlightId** Flight being used
- **Quiet** Whether Setup will run in quiet mode or in full
- **RelatedCV** Correlation vector of any other related events
- **SetupMode** Phase that Setup is in

#### **DeploymentTelemetry.Deployment\_Start**

Event to indicate that a Deployment 360 API has been called.

The following fields are available:

- **ClientId** Client ID of user utilizing the D360 API
- **FlightId** Flight being used
- **Mode** Phase in upgrade
- **RelatedCV** CV of any other related events

## Diagnostic data events

### TelClientSynthetic.AuthorizationInfo\_RuntimeTransition

Fired by UTC at state transitions to signal what data we are allowed to collect.

The following fields are available:

- **CanAddMsaToMsTelemetry** True if we can add MSA PUID and CID to telemetry, false otherwise.
- **CanCollectAnyTelemetry** True if we are allowed to collect partner telemetry, false otherwise.
- **CanCollectCoreTelemetry** True if we can collect CORE/Basic telemetry, false otherwise.
- **CanCollectHeartbeats** True if we can collect heartbeat telemetry, false otherwise.
- **CanCollectOsTelemetry** True if we can collect diagnostic data telemetry, false otherwise.
- **CanCollectWindowsAnalyticsEvents** True if we can collect Windows Analytics data, false otherwise.
- **CanPerformDiagnosticEscalations** True if we can perform diagnostic escalation collection, false otherwise.
- **CanPerformTraceEscalations** True if we can perform trace escalation collection, false otherwise.
- **CanReportScenarios** True if we can report scenario completions, false otherwise.
- **PreviousPermissions** Bitmask of previous telemetry state.
- **TransitionFromEverythingOff** True if we are transitioning from all telemetry being disabled, false otherwise.

### TelClientSynthetic.AuthorizationInfo\_Startup

Fired by UTC at startup to signal what data we are allowed to collect.

The following fields are available:

- **CanAddMsaToMsTelemetry** True if we can add MSA PUID and CID to telemetry, false otherwise.
- **CanCollectAnyTelemetry** True if we are allowed to collect partner telemetry, false otherwise.
- **CanCollectCoreTelemetry** True if we can collect CORE/Basic telemetry, false otherwise.
- **CanCollectHeartbeats** True if we can collect heartbeat telemetry, false otherwise.
- **CanCollectOsTelemetry** True if we can collect diagnostic data telemetry, false otherwise.
- **CanCollectWindowsAnalyticsEvents** True if we can collect Windows Analytics data, false otherwise.
- **CanPerformDiagnosticEscalations** True if we can perform diagnostic escalation collection, false otherwise.
- **CanPerformTraceEscalations** True if we can perform trace escalation collection, false otherwise.
- **CanReportScenarios** True if we can report scenario completions, false otherwise.
- **PreviousPermissions** Bitmask of previous telemetry state.
- **TransitionFromEverythingOff** True if we are transitioning from all telemetry being disabled, false otherwise.

### TelClientSynthetic.HeartBeat\_5

Fired by UTC as a heartbeat signal.

The following fields are available:

- **AgentConnectionErrorsCount** Number of non-timeout errors associated with the host/agent channel.
- **CensusExitCode** Last exit code of Census task.
- **CensusStartTime** Time of last Census run.
- **CensusTaskEnabled** True if Census is enabled, false otherwise.
- **CompressedBytesUploaded** Number of compressed bytes uploaded.
- **ConsumerDroppedCount** Number of events dropped at consumer layer of telemetry client.
- **CriticalDataDbDroppedCount** Number of critical data sampled events dropped at the database layer.
- **CriticalDataThrottleDroppedCount** Number of critical data sampled events dropped due to throttling.
- **CriticalOverflowEntersCounter** Number of times critical overflow mode was entered in event DB.
- **DbCriticalDroppedCount** Total number of dropped critical events in event DB.
- **DbDroppedCount** Number of events dropped due to DB fullness.
- **DbDroppedFailureCount** Number of events dropped due to DB failures.

- **DbDroppedFullCount** Number of events dropped due to DB fullness.
- **DecodingDroppedCount** Number of events dropped due to decoding failures.
- **EnteringCriticalOverflowDroppedCounter** Number of events dropped due to critical overflow mode being initiated.
- **EtwDroppedBufferCount** Number of buffers dropped in the UTC ETW session.
- **EtwDroppedCount** Number of events dropped at ETW layer of telemetry client.
- **EventsPersistedCount** Number of events that reached the PersistEvent stage.
- **EventSubStoreResetCounter** Number of times event DB was reset.
- **EventSubStoreResetSizeSum** Total size of event DB across all resets reports in this instance.
- **EventsUploaded** Number of events uploaded.
- **Flags** Flags indicating device state such as network state, battery state, and opt-in state.
- **FullTriggerBufferDroppedCount** Number of events dropped due to trigger buffer being full.
- **HeartBeatSequenceNumber** The sequence number of this heartbeat.
- **InvalidHttpCodeCount** Number of invalid HTTP codes received from contacting Vortex.
- **LastAgentConnectionError** Last non-timeout error encountered in the host/agent channel.
- **LastEventSizeOffender** Event name of last event which exceeded max event size.
- **LastInvalidHttpCode** Last invalid HTTP code received from Vortex.
- **MaxActiveAgentConnectionCount** Maximum number of active agents during this heartbeat timeframe.
- **MaxInUseScenarioCounter** Soft maximum number of scenarios loaded by UTC.
- **PreviousHeartBeatTime** Time of last heartbeat event (allows chaining of events).
- **SettingsHttpAttempts** Number of attempts to contact OneSettings service.
- **SettingsHttpFailures** Number of failures from contacting OneSettings service.
- **ThrottledDroppedCount** Number of events dropped due to throttling of noisy providers.
- **UploaderDroppedCount** Number of events dropped at the uploader layer of telemetry client.
- **VortexFailuresTimeout** Number of time out failures received from Vortex.
- **VortexHttpAttempts** Number of attempts to contact Vortex.
- **VortexHttpFailures4xx** Number of 400-499 error codes received from Vortex.
- **VortexHttpFailures5xx** Number of 500-599 error codes received from Vortex.
- **VortexHttpResponseFailures** Number of Vortex responses that are not 2XX or 400.
- **VortexHttpResponsesWithDroppedEvents** Number of Vortex responses containing at least 1 dropped event.
- **EventStoreLifetimeResetCounter** Number of times event DB was reset for the lifetime of UTC.
- **EventStoreResetCounter** Number of times event DB was reset.
- **EventStoreResetSizeSum** Total size of event DB across all resets reports in this instance.

#### TelClientSynthetic.HeartBeat\_Aria\_5

Telemetry client ARIA heartbeat event.

The following fields are available:

- **CompressedBytesUploaded** Number of compressed bytes uploaded.
- **CriticalDataDbDroppedCount** Number of critical data sampled events dropped at the database layer.
- **CriticalOverflowEntersCounter** Number of times critical overflow mode was entered in event DB.
- **DbCriticalDroppedCount** Total number of dropped critical events in event DB.
- **DbDroppedCount** Number of events dropped at the DB layer.
- **DbDroppedFailureCount** Number of events dropped due to DB failures.
- **DbDroppedFullCount** Number of events dropped due to DB fullness.
- **EnteringCriticalOverflowDroppedCounter** Number of events dropped due to critical overflow mode being

initiated.

- **EventsPersistedCount** Number of events that reached the PersistEvent stage.
- **EventSubStoreResetCounter** Number of times event DB was reset.
- **EventSubStoreResetSizeSum** Total size of event DB across all resets reports in this instance.
- **EventsUploaded** Number of events uploaded.
- **HeartBeatSequenceNumber** The sequence number of this heartbeat.
- **InvalidHttpCodeCount** Number of invalid HTTP codes received from contacting Vortex.
- **LastEventSizeOffender** Event name of last event which exceeded max event size.
- **LastInvalidHttpCode** Last invalid HTTP code received from Vortex.
- **PreviousHeartBeatTime** The FILETIME of the previous heartbeat fire.
- **SettingsHttpAttempts** Number of attempts to contact OneSettings service.
- **SettingsHttpFailures** Number of failures from contacting OneSettings service.
- **UploaderDroppedCount** Number of events dropped at the uploader layer of telemetry client.
- **VortexFailuresTimeout** Number of time out failures received from Vortex.
- **VortexHttpAttempts** Number of attempts to contact Vortex.
- **VortexHttpFailures4xx** Number of 400-499 error codes received from Vortex.
- **VortexHttpFailures5xx** Number of 500-599 error codes received from Vortex.
- **VortexHttpResponseFailures** Number of Vortex responses that are not 2XX or 400.
- **VortexHttpResponsesWithDroppedEvents** Number of Vortex responses containing at least 1 dropped event.

#### **TelClientSynthetic.PrivacySettingsAfterCreatorsUpdate**

This event sends basic data on privacy settings before and after a feature update. This is used to ensure that customer privacy settings are correctly migrated across feature updates.

The following fields are available:

- **PostUpgradeSettings** The privacy settings after a feature update.
- **PreUpgradeSettings** The privacy settings before a feature update.

## Direct to update events

#### **Microsoft.Windows.DirectToUpdate.DTUCoordinatorCheckApplicability**

Event to indicate that the Coordinator CheckApplicability call succeeded.

The following fields are available:

- **ApplicabilityResult** Result of CheckApplicability function.
- **CampaignID** Campaign ID being run.
- **ClientID** Client ID being run.
- **CoordinatorVersion** Coordinator version of DTU.
- **CV** Correlation vector.

#### **Microsoft.Windows.DirectToUpdate.DTUCoordinatorCheckApplicabilityGenericFailure**

Event to indicate that we have received an unexpected error in the DTU Coordinators CheckApplicability call.

The following fields are available:

- **hResult** HRESULT of the failure.
- **CampaignID** Campaign ID being run.
- **ClientID** Client ID being run.
- **CoordinatorVersion** Coordinator version of DTU.



- **CV** Correlation vector.

#### **Microsoft.Windows.DirectToUpdate.DTUCoordinatorCommitGenericFailure**

Commit call.

The following fields are available:

- **hResult** HRESULT of the failure.
- **CampaignID** Campaign ID being run.
- **ClientID** Client ID being run.
- **CoordinatorVersion** Coordinator version of DTU.
- **CV** Correlation vector.

#### **Microsoft.Windows.DirectToUpdate.DTUCoordinatorCommitSuccess**

Event to indicate that the Coordinator Commit call succeeded.

The following fields are available:

- **CampaignID** Campaign ID being run.
- **ClientID** Client ID being run.
- **CoordinatorVersion** Coordinator version of DTU.
- **CV** Correlation vector.

#### **Microsoft.Windows.DirectToUpdate.DTUCoordinatorDownloadGenericFailure**

Event to indicate that we have received an unexpected error in the DTU Coordinator Download call.

The following fields are available:

- **CampaignID** Campaign ID being run.
- **ClientID** Client ID being run.
- **CoordinatorVersion** Coordinator version of DTU.
- **CV** Correlation vector.
- **hResult** HRESULT of the failure.

#### **Microsoft.Windows.DirectToUpdate.DTUCoordinatorDownloadIgnoredFailure**

Event to indicate that we have received an error in the DTU Coordinator Download call that will be ignored.

The following fields are available:

- **CampaignID** Campaign ID being run.
- **ClientID** Client ID being run.
- **CoordinatorVersion** Coordinator version of DTU.
- **CV** Correlation vector.
- **hResult** HRESULT of the failure.

#### **Microsoft.Windows.DirectToUpdate.DTUCoordinatorDownloadSuccess**

Event to indicate that the Coordinator Download call succeeded.

The following fields are available:

- **CampaignID** Campaign ID being run.
- **ClientID** Client ID being run.
- **CoordinatorVersion** Coordinator version of DTU.
- **CV** Correlation vector.

#### **Microsoft.Windows.DirectToUpdate.DTUCoordinatorHandleShutdownGenericFailure**

Event to indicate that we have received an unexpected error in the DTU Coordinator HandleShutdown call.

The following fields are available:

- **CampaignID** Campaign ID being run.
- **ClientID** Client ID being run.
- **CoordinatorVersion** Coordinate version of DTU.
- **CV** Correlation vector.
- **hResult** HRESULT of the failure.

#### **Microsoft.Windows.DirectToUpdate.DTUCoordinatorHandleShutdownSuccess**

Event to indicate that the Coordinator HandleShutdown call succeeded.

The following fields are available:

- **CampaignID** Campaign ID being run.
- **ClientID** Client ID being run.
- **CoordinatorVersion** Coordinator version of DTU.
- **CV** Correlation vector.

#### **Microsoft.Windows.DirectToUpdate.DTUCoordinatorInitializeGenericFailure**

Event to indicate that we have received an unexpected error in the DTU Coordinator Initialize call.

The following fields are available:

- **hResult** HRESULT of the failure.
- **CampaignID** Campaign ID being run.
- **ClientID** Client ID being run.
- **CoordinatorVersion** Coordinator version of DTU.
- **CV** Correlation vector.

#### **Microsoft.Windows.DirectToUpdate.DTUCoordinatorInitializeSuccess**

Event to indicate that the Coordinator Initialize call succeeded.

The following fields are available:

- **CampaignID** Campaign ID being run.
- **ClientID** Client ID being run.
- **CoordinatorVersion** Coordinator version of DTU.
- **CV** Correlation vector.

#### **Microsoft.Windows.DirectToUpdate.DTUCoordinatorInstallGenericFailure**

Event to indicate that we have received an unexpected error in the DTU Coordinator Install call.

The following fields are available:

- **CampaignID** Campaign ID being run.
- **ClientID** Client ID being run.
- **CoordinatorVersion** Coordinator version of DTU.
- **CV** Correlation vector.
- **hResult** HRESULT of the failure.

#### **Microsoft.Windows.DirectToUpdate.DTUCoordinatorInstallIgnoredFailure**

Event to indicate that we have received an error in the DTU Coordinator Install call that will be ignored.

The following fields are available:

- **CampaignID** Campaign ID being run.
- **ClientID** Client ID being run.
- **CoordinatorVersion** Coordinator version of DTU.
- **CV** Correlation vector.
- **hResult** HRESULT of the failure.

#### **Microsoft.Windows.DirectToUpdate.DTUCoordinatorInstallSuccess**

Event to indicate that the Coordinator Install call succeeded.

The following fields are available:

- **CampaignID** Campaign ID being run.
- **ClientID** Client ID being run.
- **CoordinatorVersion** Coordinator version of DTU.
- **CV** Correlation vector.

#### **Microsoft.Windows.DirectToUpdate.DTUCoordinatorProgressCallBack**

Event to indicate Coordinator's progress callback has been called.

The following fields are available:

- **Current Deploy Phase's percentage completed** Trigger which fired UXLauncher.
- **DeployPhase** Current Deploy Phase.
- **CampaignID** Campaign ID being run.
- **ClientID** Client ID being run.
- **CoordinatorVersion** Coordinator version of DTU.
- **CV** Correlation vector.

#### **Microsoft.Windows.DirectToUpdate.DTUCoordinatorSetCommitReadyGenericFailure**

Event to indicate that we have received an unexpected error in the DTU Coordinator SetCommitReady call.

The following fields are available:

- **CampaignID** Campaign ID being run.
- **ClientID** Client ID being run.
- **CoordinatorVersion** Coordinator version of DTU.
- **CV** Correlation vector.
- **hResult** HRESULT of the failure.

#### **Microsoft.Windows.DirectToUpdate.DTUCoordinatorSetCommitReadySuccess**

Event to indicate that the Coordinator SetCommitReady call succeeded.

The following fields are available:

- **CampaignID** Campaign ID being run.
- **ClientID** Client ID being run.
- **CoordinatorVersion** Coordinator version of DTU.
- **CV** Correlation vector.

#### **Microsoft.Windows.DirectToUpdate.DTUCoordinatorWaitForRebootUiGenericFailure**

Event to indicate that we have received an unexpected error in the DTU Coordinator WaitForRebootUi call.

The following fields are available:

- **CampaignID** Campaign ID being run.

- **ClientID** Client ID being run.
- **CoordinatorVersion** Coordinator version of DTU.
- **CV** Correlation vector.
- **hResult** HRESULT of the failure.

#### **Microsoft.Windows.DirectToUpdate.DTUCoordinatorWaitForRebootUiNotShown**

Event to indicate that the Coordinator WaitForRebootUi call succeeded.

The following fields are available:

- **CampaignID** Campaign ID being run
- **ClientID** Client ID being run
- **CoordinatorVersion** Coordinator version of DTU
- **CV** Correlation vector
- **hResult** HRESULT of the failure

#### **Microsoft.Windows.DirectToUpdate.DTUCoordinatorWaitForRebootUiSelection**

Event to indicate the user selected an option on the Reboot UI.

The following fields are available:

- **CampaignID** Campaign ID being run
- **ClientID** Client ID being run
- **CoordinatorVersion** Coordinator version of DTU
- **CV** Correlation vector
- **rebootUiSelection** Selection on the Reboot UI

#### **Microsoft.Windows.DirectToUpdate.DTUCoordinatorWaitForRebootUiSuccess**

Event to indicate that the Coordinator WaitForRebootUi call succeeded.

The following fields are available:

- **CampaignID** Campaign ID being run
- **ClientID** Client ID being run
- **CoordinatorVersion** Coordinator version of DTU
- **CV** Correlation vector

#### **Microsoft.Windows.DirectToUpdate.DTUHandlerCheckApplicabilityGenericFailure**

Event to indicate that we have received an unexpected error in the DTU Handler CheckApplicability call.

The following fields are available:

- **hResult** HRESULT of the failure
- **CampaignID** Campaign ID being run
- **ClientID** Client ID being run
- **CoordinatorVersion** Coordinator version of DTU
- **CV** Correlation vector
- **CV\_new** New correlation vector

#### **Microsoft.Windows.DirectToUpdate.DTUHandlerCheckApplicabilityInternalGenericFailure**

Event to indicate that we have received an unexpected error in the DTU Handler CheckApplicabilityInternal call.

The following fields are available:

- **CampaignID** Campaign ID being run

- **ClientID** Client ID being run
- **CoordinatorVersion** Coordinator version of DTU
- **CV** Correlation vector
- **hResult** HRESULT of the failure

#### **Microsoft.Windows.DirectToUpdate.DTUHandlerCheckApplicabilityInternalSuccess**

Event to indicate that the Handler CheckApplicabilityInternal call succeeded.

The following fields are available:

- **ApplicabilityResult** Result of CheckApplicability function
- **CampaignID** Campaign ID being run
- **ClientID** Client ID being run
- **CoordinatorVersion** Coordinator version of DTU
- **CV** Correlation vector

#### **Microsoft.Windows.DirectToUpdate.DTUHandlerCheckApplicabilitySuccess**

Event to indicate that the Handler CheckApplicability call succeeded.

The following fields are available:

- **ApplicabilityResult** Result of CheckApplicability function
- **CampaignID** Campaign ID being run
- **ClientID** Client ID being run
- **CoordinatorVersion** Coordinator version of DTU
- **CV** Correlation vector
- **CV\_new** New correlation vector

#### **Microsoft.Windows.DirectToUpdate.DTUHandlerCheckIfCoordinatorMinApplicableVersionGenericFailure**

Event to indicate that we have received an unexpected error in the DTU Handler CheckIfCoordinatorMinApplicableVersion call.

The following fields are available:

- **CampaignID** Campaign ID being run
- **ClientID** Client ID being run
- **CoordinatorVersion** Coordinator version of DTU
- **CV** Correlation vector
- **hResult** HRESULT of the failure

#### **Microsoft.Windows.DirectToUpdate.DTUHandlerCheckIfCoordinatorMinApplicableVersionSuccess**

Event to indicate that the Handler CheckIfCoordinatorMinApplicableVersion call succeeded.

The following fields are available:

- **CampaignID** Campaign ID being run
- **CheckIfCoordinatorMinApplicableVersionResult** Result of CheckIfCoordinatorMinApplicableVersion function
- **ClientID** Client ID being run
- **CoordinatorVersion** Coordinator version of DTU
- **CV** Correlation vector

#### **Microsoft.Windows.DirectToUpdate.DTUHandlerCommitGenericFailure**

Event to indicate that we have received an unexpected error in the DTU Handler Commit call.

The following fields are available:

- **CampaignID** Campaign ID being run
- **ClientID** Client ID being run
- **CoordinatorVersion** Coordinator version of DTU
- **CV** Correlation vector
- **CV\_new** New correlation vector
- **hResult** HRESULT of the failure

#### **Microsoft.Windows.DirectToUpdate.DTUHandlerCommitSuccess**

Event to indicate that the Handler Commit call succeeded.

The following fields are available:

- **CampaignID** Campaign ID being run
- **ClientID** Client ID being run
- **CoordinatorVersion** Coordinator version of DTU
- **CV** Correlation vector
- **CV\_new** New correlation vector

#### **Microsoft.Windows.DirectToUpdate.DTUHandlerDownloadAndExtractCabAlreadyDownloaded**

Event to indicate that the Handler Download and Extract cab returned a value indicating that the cab trying to be downloaded has already been downloaded.

The following fields are available:

- **CampaignID** Campaign ID being run
- **ClientID** Client ID being run
- **CoordinatorVersion** Coordinator version of DTU
- **CV** Correlation vector

#### **Microsoft.Windows.DirectToUpdate.DTUHandlerDownloadAndExtractCabFailure**

Event to indicate that the Handler Download and Extract cab call failed.

The following fields are available:

- **CampaignID** Campaign ID being run
- **ClientID** Client ID being run
- **CoordinatorVersion** Coordinator version of DTU
- **CV** Correlation vector
- **DownloadAndExtractCabFunction\_failureReason** Reason why the DownloadAndExtractCab function failed
- **hResult** HRESULT of the failure

#### **Microsoft.Windows.DirectToUpdate.DTUHandlerDownloadAndExtractCabSuccess**

Event to indicate that the Handler Download and Extract cab call succeeded.

The following fields are available:

- **CampaignID** Campaign ID being run
- **ClientID** Client ID being run
- **CoordinatorVersion** Coordinator version of DTU
- **CV** Correlation vector

#### **Microsoft.Windows.DirectToUpdate.DTUHandlerDownloadGenericFailure**

Event to indicate that we have received an unexpected error in the DTU Handler Download call.

The following fields are available:

- **CampaignID** Campaign ID being run
- **ClientID** Client ID being run
- **CoordinatorVersion** Coordinator version of DTU
- **CV** Correlation vector
- **hResult** HRESULT of the failure

#### **Microsoft.Windows.DirectToUpdate.DTUHandlerDownloadSuccess**

Event to indicate that the Handler Download call succeeded.

The following fields are available:

- **CampaignID** Campaign ID being run
- **ClientID** Client ID being run
- **CoordinatorVersion** Coordinator version of DTU
- **CV** Correlation vector

#### **Microsoft.Windows.DirectToUpdate.DTUHandlerInitializeGenericFailure**

Event to indicate that we have received an unexpected error in the DTU Handler Initialize call.

The following fields are available:

- **CampaignID** Campaign ID being run
- **ClientID** Client ID being run
- **CoordinatorVersion** Coordinator version of DTU
- **CV** Correlation vector
- **DownloadAndExtractCabFunction\_hResult** HRESULT of the DownloadAndExtractCab function
- **hResult** HRESULT of the failure

#### **Microsoft.Windows.DirectToUpdate.DTUHandlerInitializeSuccess**

Event to indicate that the Handler Initialize call succeeded.

The following fields are available:

- **CampaignID** Campaign ID being run
- **ClientID** Client ID being run
- **CoordinatorVersion** Coordinator version of DTU
- **CV** Correlation vector
- **DownloadAndExtractCabFunction\_hResult** HRESULT of the DownloadAndExtractCab function

#### **Microsoft.Windows.DirectToUpdate.DTUHandlerInstallGenericFailure**

Event to indicate that we have received an unexpected error in the DTU Handler Install call.

The following fields are available:

- **CampaignID** Campaign ID being run
- **ClientID** Client ID being run
- **CoordinatorVersion** Coordinator version of DTU
- **CV** Correlation vector
- **hResult** HRESULT of the failure

#### **Microsoft.Windows.DirectToUpdate.DTUHandlerInstallSuccess**

Event to indicate that the Coordinator Install call succeeded.

The following fields are available:

- **CampaignID** Campaign ID being run
- **ClientID** Client ID being run
- **CoordinatorVersion** Coordinator version of DTU
- **CV** Correlation vector

#### **Microsoft.Windows.DirectToUpdate.DTUHandlerSetCommitReadyGenericFailure**

Event to indicate that we have received an unexpected error in the DTU Handler SetCommitReady call.

The following fields are available:

- **hResult** HRESULT of the failure
- **CampaignID** Campaign ID being run
- **ClientID** Client ID being run
- **CoordinatorVersion** Coordinator version of DTU
- **CV** Correlation vector

#### **Microsoft.Windows.DirectToUpdate.DTUHandlerSetCommitReadySuccess**

Event to indicate that the Handler SetCommitReady call succeeded.

The following fields are available:

- **CampaignID** Campaign ID being run
- **ClientID** Client ID being run
- **CoordinatorVersion** Coordinator version of DTU
- **CV** Correlation vector

#### **Microsoft.Windows.DirectToUpdate.DTUHandlerWaitForRebootUiGenericFailure**

Event to indicate that we have received an unexpected error in the DTU Handler WaitForRebootUi call.

The following fields are available:

- **hResult** HRESULT of the failure
- **CampaignID** Campaign ID being run
- **ClientID** Client ID being run
- **CoordinatorVersion** Coordinator version of DTU
- **CV** Correlation vector

#### **Microsoft.Windows.DirectToUpdate.DTUHandlerWaitForRebootUiSuccess**

Event to indicate that the Handler WaitForRebootUi call succeeded.

The following fields are available:

- **CampaignID** Campaign ID being run
- **ClientID** Client ID being run
- **CoordinatorVersion** Coordinator version of DTU
- **CV** Correlation vector

## Feature update events

#### **Microsoft.Windows.Upgrade.Uninstall.UninstallFailed**

This event sends diagnostic data about failures when uninstalling a feature update, to help resolve any issues preventing customers from reverting to a known state



The following fields are available:

- **failureReason** Provides data about the uninstall initialization operation failure
- **hr** Provides the Win32 error code for the operation failure

#### **Microsoft.Windows.Upgrade.Uninstall.UninstallFinalizedAndRebootTriggered**

Indicates that the uninstall was properly configured and that a system reboot was initiated

#### **Microsoft.Windows.Upgrade.Uninstall.UninstallGoBackButtonClicked**

This event sends basic metadata about the starting point of uninstalling a feature update which helps us ensure customers can safely revert to a well-known state if the update caused any problems.

## Inventory events

#### **Microsoft.Windows.Inventory.Core.AmiTelCacheChecksum**

This event captures basic checksum data about the device inventory items stored in the cache for use in validating data completeness for Microsoft.Windows.Inventory.Core events. The fields in this event may change over time, but they will always represent a count of a given object.

The following fields are available:

- **DriverPackageExtended** A count of driverpackageextended objects in cache
- **FileSigningInfo** A count of file signing objects in cache
- **InventoryApplication** A count of application objects in cache
- **InventoryApplicationFile** A count of application file objects in cache
- **InventoryDeviceContainer** A count of device container objects in cache
- **InventoryDeviceInterface** A count of PNP device interface objects in cache
- **InventoryDeviceMediaClass** A count of device media objects in cache
- **InventoryDevicePnp** A count of devicepnp objects in cache
- **InventoryDeviceUsbHubClass** A count of device usb objects in cache
- **InventoryDriverBinary** A count of driver binary objects in cache
- **InventoryDriverPackage** A count of device objects in cache

#### **Microsoft.Windows.Inventory.Core.AmiTelCacheVersions**

This event sends inventory component versions for the Device Inventory data.

The following fields are available:

- **aeinv** The version of the App inventory component.
- **devinv** The file version of the Device inventory component.

#### **Microsoft.Windows.Inventory.Core.InventoryApplicationAdd**

This event sends basic metadata about an application on the system to help keep Windows up to date.

The following fields are available:

- **HiddenArp** Indicates whether a program hides itself from showing up in ARP.
- **InstallDate** The date the application was installed (a best guess based on folder creation date heuristics).
- **InstallDateArpLastModified** The date of the registry ARP key for a given application. Hints at install date but not always accurate. Passed as an array. Example: 4/11/2015 00:00:00
- **InstallDateFromLinkFile** The estimated date of install based on the links to the files. Passed as an array.
- **InstallDateMsi** The install date if the application was installed via MSI. Passed as an array.
- **InventoryVersion** The version of the inventory file generating the events.
- **Language** The language code of the program.

- **MsiPackageCode** A GUID that describes the MSI Package. Multiple 'Products' (apps) can make up an MsiPackage.
- **MsiProductCode** A GUID that describe the MSI Product.
- **Name** The name of the application
- **OSVersionAtInstallTime** The four octets from the OS version at the time of the application's install.
- **PackageFullName** The package full name for a Store application.
- **ProgramInstanceId** A hash of the file IDs in an app.
- **Publisher** The Publisher of the application. Location pulled from depends on the 'Source' field.
- **RootDirPath** The path to the root directory where the program was installed.
- **Source** How the program was installed (ARP, MSI, Appx, etc...)
- **StoreAppType** A sub-classification for the type of Microsoft Store app, such as UWP or Win8StoreApp.
- **Type** One of ("Application", "Hotfix", "BOE", "Service", "Unknown"). Application indicates Win32 or Appx app, Hotfix indicates app updates (KBs), BOE indicates it's an app with no ARP or MSI entry, Service indicates that it is a service. Application and BOE are the ones most likely seen.
- **Version** The version number of the program.

#### **Microsoft.Windows.Inventory.Core.InventoryApplicationFrameworkAdd**

This event provides the basic metadata about the frameworks an application may depend on

The following fields are available:

- **FileId** A hash that uniquely identifies a file
- **Frameworks** The list of frameworks this file depends on
- **InventoryVersion** The version of the inventory file generating the events

#### **Microsoft.Windows.Inventory.Core.InventoryApplicationFrameworkStartSync**

This event indicates that a new set of InventoryApplicationFrameworkAdd events will be sent

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events

#### **Microsoft.Windows.Inventory.Core.InventoryApplicationRemove**

This event indicates that a new set of InventoryDevicePnpAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryApplicationStartSync**

This event indicates that a new set of InventoryApplicationAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDeviceContainerAdd**

This event sends basic metadata about a device container (such as a monitor or printer as opposed to a PNP device) to help keep Windows up-to-date.

The following fields are available:

- **Categories** A comma separated list of functional categories in which the container belongs.
- **DiscoveryMethod** The discovery method for the device container.
- **FriendlyName** The name of the device container.
- **InventoryVersion** The version of the inventory file generating the events.

- **IsActive** Is the device connected, or has it been seen in the last 14 days?
- **IsConnected** For a physically attached device, this value is the same as IsPresent. For wireless a device, this value represents a communication link.
- **IsMachineContainer** Is the container the root device itself?
- **IsNetworked** Is this a networked device?
- **IsPaired** Does the device container require pairing?
- **Manufacturer** The manufacturer name for the device container.
- **ModelId** A model GUID.
- **ModelName** The model name.
- **ModelNumber** The model number for the device container.
- **PrimaryCategory** The primary category for the device container.

#### **Microsoft.Windows.Inventory.Core.InventoryDeviceContainerRemove**

This event indicates that the InventoryDeviceContainer object is no longer present.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDeviceContainerStartSync**

This event indicates that a new set of InventoryDeviceContainerAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDeviceInterfaceAdd**

This event retrieves information about what sensor interfaces are available on the device.

The following fields are available:

- **Accelerometer3D** Indicates if an Accelerator3D sensor is found.
- **ActivityDetection** Indicates if an Activity Detection sensor is found.
- **AmbientLight** Indicates if an Ambient Light sensor is found.
- **Barometer** Indicates if a Barometer sensor is found.
- **Custom** Indicates if a Custom sensor is found.
- **EnergyMeter** Indicates if an Energy sensor is found.
- **FloorElevation** Indicates if a Floor Elevation sensor is found.
- **GeomagneticOrientation** Indicates if a Geo Magnetic Orientation sensor is found.
- **GravityVector** Indicates if a Gravity Detector sensor is found.
- **Gyrometer3D** Indicates if a Gyrometer3D sensor is found.
- **Humidity** Indicates if a Humidity sensor is found.
- **InventoryVersion** The version of the inventory file generating the events.
- **LinearAccelerometer** Indicates if a Linear Accelerometer sensor is found.
- **Magnetometer3D** Indicates if a Magnetometer3D sensor is found.
- **Orientation** Indicates if an Orientation sensor is found.
- **Pedometer** Indicates if a Pedometer sensor is found.
- **Proximity** Indicates if a Proximity sensor is found.
- **RelativeOrientation** Indicates if a Relative Orientation sensor is found.
- **SimpleDeviceOrientation** Indicates if a Simple Device Orientation sensor is found.
- **Temperature** Indicates if a Temperature sensor is found.

### Microsoft.Windows.Inventory.Core.InventoryDeviceInterfaceStartSync

This event indicates that a new set of InventoryDeviceInterfaceAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

### Microsoft.Windows.Inventory.Core.InventoryDeviceMediaClassAdd

This event sends additional metadata about a PNP device that is specific to a particular class of devices to help keep Windows up to date while reducing overall size of data payload.

The following fields are available:

- **Audio\_CaptureDriver** The Audio device capture driver endpoint.
- **Audio\_RenderDriver** The Audio device render driver endpoint.
- **InventoryVersion** The version of the inventory file generating the events.

### Microsoft.Windows.Inventory.Core.InventoryDeviceMediaClassStartSync

This event indicates that a new set of InventoryDeviceMediaClassSAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

### Microsoft.Windows.Inventory.Core.InventoryDevicePnpAdd

This event represents the basic metadata about a PNP device and its associated driver

The following fields are available:

- **BusReportedDescription** System-supplied GUID that uniquely groups the functional devices associated with a single-function or multifunction device installed in the computer.
- **Class** A unique identifier for the driver installed.
- **ClassGuid** Name of the .sys image file (or wudfrd.sys if using user mode driver framework).
- **COMPID** INF file name (the name could be renamed by OS, such as oemXX.inf)
- **ContainerId** The version of the inventory binary generating the events.
- **Description** The current error code for the device.
- **DeviceState** The device description.
- **DriverId** DeviceState is a bitmask of the following: DEVICE\_IS\_CONNECTED 0x0001 (currently only for container). DEVICE\_IS\_NETWORK\_DEVICE 0x0002 (currently only for container). DEVICE\_IS\_PAIRED 0x0004 (currently only for container). DEVICE\_IS\_ACTIVE 0x0008 (currently never set). DEVICE\_IS\_MACHINE 0x0010 (currently only for container). DEVICE\_IS\_PRESENT 0x0020 (currently always set). DEVICE\_IS\_HIDDEN 0x0040. DEVICE\_IS\_PRINTER 0x0080 (currently only for container). DEVICE\_IS\_WIRELESS 0x0100. DEVICE\_IS\_WIRELESS\_FAT 0x0200. The most common values are therefore: 32 (0x20)= device is present. 96 (0x60)= device is present but hidden. 288 (0x120)= device is a wireless device that is present
- **DriverName** A unique identifier for the driver installed.
- **DriverPackageStrongName** The immediate parent directory name in the Directory field of InventoryDriverPackage
- **DriverVerDate** Name of the .sys image file (or wudfrd.sys if using user mode driver framework).
- **DriverVerVersion** The immediate parent directory name in the Directory field of InventoryDriverPackage.
- **Enumerator** The date of the driver loaded for the device.
- **HWID** The version of the driver loaded for the device.
- **Inf** The bus that enumerated the device.
- **InstallState** The device installation state. One of these values: <https://msdn.microsoft.com/en-us/library/windows/hardware/ff543130.aspx>

- **InventoryVersion** List of hardware ids for the device.
- **LowerClassFilters** Lower filter class drivers IDs installed for the device
- **LowerFilters** Lower filter drivers IDs installed for the device
- **Manufacturer** INF file name (the name could be renamed by OS, such as oemXX.inf)
- **MatchingID** Device installation state.
- **Model** The version of the inventory binary generating the events.
- **ParentId** Lower filter class drivers IDs installed for the device.
- **ProblemCode** Lower filter drivers IDs installed for the device.
- **Provider** The device manufacturer.
- **Service** The device service name
- **STACKID** Represents the hardware ID or compatible ID that Windows uses to install a device instance.
- **UpperClassFilters** Upper filter drivers IDs installed for the device
- **UpperFilters** The device model.

#### **Microsoft.Windows.Inventory.Core.InventoryDevicePnpRemove**

This event indicates that the InventoryDevicePnpRemove object is no longer present.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDevicePnpStartSync**

This event indicates that a new set of InventoryDevicePnpAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDeviceUsbHubClassAdd**

This event sends basic metadata about the USB hubs on the device

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events
- **TotalUserConnectablePorts** Total number of connectable USB ports
- **TotalUserConnectableTypeCPorts** Total number of connectable USB Type C ports

#### **Microsoft.Windows.Inventory.Core.InventoryDeviceUsbHubClassStartSync**

This event indicates that a new set of InventoryDeviceUsbHubClassAdd events will be sent

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events

#### **Microsoft.Windows.Inventory.Core.InventoryDriverBinaryAdd**

This event provides the basic metadata about driver binaries running on the system

The following fields are available:

- **DriverChecksum** The checksum of the driver file.
- **DriverCompany** The company name that developed the driver.
- **DriverInBox** Is the driver included with the operating system?
- **DriverIsKernelMode** Is it a kernel mode driver?
- **DriverName** The file name of the driver.
- **DriverPackageStrongName** The strong name of the driver package

- **DriverSigned** The strong name of the driver package
- **DriverTimeStamp** The low 32 bits of the time stamp of the driver file.
- **DriverType** A bitfield of driver attributes: 1. define DRIVER\_MAP\_DRIVER\_TYPE\_PRINTER 0x0001. 2. define DRIVER\_MAP\_DRIVER\_TYPE\_KERNEL 0x0002. 3. define DRIVER\_MAP\_DRIVER\_TYPE\_USER 0x0004. 4. define DRIVER\_MAP\_DRIVER\_IS\_SIGNED 0x0008. 5. define DRIVER\_MAP\_DRIVER\_IS\_INBOX 0x0010. 6. define DRIVER\_MAP\_DRIVER\_IS\_WINQUAL 0x0040. 7. define DRIVER\_MAP\_DRIVER\_IS\_SELF\_SIGNED 0x0020. 8. define DRIVER\_MAP\_DRIVER\_IS\_CI\_SIGNED 0x0080. 9. define DRIVER\_MAP\_DRIVER\_HAS\_BOOT\_SERVICE 0x0100. 10. define DRIVER\_MAP\_DRIVER\_TYPE\_I386 0x10000. 11. define DRIVER\_MAP\_DRIVER\_TYPE\_IA64 0x20000. 12. define DRIVER\_MAP\_DRIVER\_TYPE\_AMD64 0x40000. 13. define DRIVER\_MAP\_DRIVER\_TYPE\_ARM 0x100000. 14. define DRIVER\_MAP\_DRIVER\_TYPE\_THUMB 0x200000. 15. define DRIVER\_MAP\_DRIVER\_TYPE\_ARMNT 0x400000. 16. define DRIVER\_MAP\_DRIVER\_IS\_TIME\_STAMPED 0x800000.
- **DriverVersion** The version of the driver file.
- **ImageSize** The size of the driver file.
- **Inf** The name of the INF file.
- **InventoryVersion** The version of the inventory file generating the events.
- **Product** The product name that is included in the driver file.
- **ProductVersion** The product version that is included in the driver file.
- **Service** The device service name
- **WdfVersion** The Windows Driver Framework version.

#### **Microsoft.Windows.Inventory.Core.InventoryDriverBinaryRemove**

This event indicates that the InventoryDriverBinary object is no longer present.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDriverBinaryStartSync**

This event indicates that a new set of InventoryDriverBinaryAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDriverPackageAdd**

This event sends basic metadata about drive packages installed on the system to help keep Windows up-to-date.

The following fields are available:

- **Class** The class name for the device driver.
- **ClassGuid** The class GUID for the device driver.
- **Date** The driver package date.
- **Directory** The path to the driver package.
- **DriverInBox** Is the driver included with the operating system?
- **Inf** The INF name of the driver package.
- **InventoryVersion** The version of the inventory file generating the events.
- **Provider** The provider for the driver package.
- **SubmissionId** The HLK submission ID for the driver package.
- **Version** The version of the driver package.

#### **Microsoft.Windows.Inventory.Core.InventoryDriverPackageRemove**

This event indicates that the InventoryDriverPackageRemove object is no longer present.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDriverPackageStartSync**

This event indicates that a new set of InventoryDriverPackageAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeAddInAdd**

Provides data on the installed Office Add-ins

The following fields are available:

- **AddInCLSID** CLSID key for the office addin
- **AddInId** Office addin ID
- **BinFileTimestamp** Timestamp of the Office addin
- **BinFileVersion** Version of the Office addin
- **Description** Office addin description
- **FileId** FileId of the Office addin
- **FriendlyName** Friendly name for office addin
- **FullPath** Unexpanded path to the office addin
- **LoadBehavior** Uint32 that describes the load behavior
- **LoadTime** Load time for the office addin
- **OfficeApplication** The office application for this addin
- **OfficeArchitecture** Architecture of the addin
- **OfficeVersion** The office version for this addin
- **OutlookCrashingAddin** Boolean that indicates if crashes have been found for this addin
- **Provider** Name of the provider for this addin

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeAddInRemove**

Indicates that this particular data object represented by the objectInstanceId is no longer present.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeAddInStartSync**

This event indicates that a new sync is being generated for this object type.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeIESettingsAdd**

This event includes the Office-related Internet Explorer features

The following fields are available:

- **OleFeatureAddon** Flag indicating which Microsoft Office products have this setting enabled. The FEATURE\_ADDON\_MANAGEMENT feature lets applications hosting the WebBrowser Control to respect add-on management selections made using the Add-on Manager feature of Internet Explorer. Add-ons disabled by the user or by administrative group policy will also be disabled in applications that enable this feature.
- **OleMachineLockdown** Flag indicating which Microsoft Office products have this setting enabled. When the FEATURE\_LOCALMACHINE\_LOCKDOWN feature is enabled, Internet Explorer applies security restrictions on content loaded from the user's local machine, which helps prevent malicious behavior involving local files.
- **OleMimeHandling** Flag indicating which Microsoft Office products have this setting enabled. When the FEATURE\_MIME\_HANDLING feature control is enabled, Internet Explorer handles MIME types more securely. Only applies to Windows Internet Explorer 6 for Windows XP Service Pack 2 (SP2)

- **OleMimeSniffing** Flag indicating which Microsoft Office products have this setting enabled. Determines a file's type by examining its bit signature. Windows Internet Explorer uses this information to determine how to render the file. The FEATURE\_MIME\_SNIFFING feature, when enabled, allows to be set differently for each security zone by using the URLACTION\_FEATURE\_MIME\_SNIFFING URL action flag
- **OleNoAxInstall** Flag indicating which Microsoft Office products have this setting enabled. When a webpage attempts to load or install an ActiveX control that isn't already installed, the FEATURE\_RESTRICT\_ACTIVEXINSTALL feature blocks the request. When a webpage tries to load or install an ActiveX control that isn't already installed, the FEATURE\_RESTRICT\_ACTIVEXINSTALL feature blocks the request
- **OleNoDownload** Flag indicating which Microsoft Office products have this setting enabled. The FEATURE\_RESTRICT\_FILEDOWNLOAD feature blocks file download requests that navigate to a resource, that display a file download dialog box, or that are not initiated explicitly by a user action (for example, a mouse click or key press). Only applies to Windows Internet Explorer 6 for Windows XP Service Pack 2 (SP2)
- **OleObjectCaching** Flag indicating which Microsoft Office products have this setting enabled. When enabled, the FEATURE\_OBJECT\_CACHING feature prevents webpages from accessing or instantiating ActiveX controls cached from different domains or security contexts
- **OlePasswordDisable** Flag indicating which Microsoft Office products have this setting enabled. After Windows Internet Explorer 6 for Windows XP Service Pack 2 (SP2), Internet Explorer no longer allows usernames and passwords to be specified in URLs that use the HTTP or HTTPS protocols. URLs using other protocols, such as FTP, still allow usernames and passwords
- **OleSafeBind** Flag indicating which Microsoft Office products have this setting enabled. The FEATURE\_SAFE\_BINDTOOBJECT feature performs additional safety checks when calling MonikerBindToObject to create and initialize Microsoft ActiveX controls. Specifically, prevent the control from being created if COMPAT\_EVIL\_DONT\_LOAD is in the registry for the control
- **OleSecurityBand** Flag indicating which Microsoft Office products have this setting enabled. The FEATURE\_SECURITYBAND feature controls the display of the Internet Explorer Information bar. When enabled, the Information bar appears when file download or code installation is restricted
- **OleUncSaveCheck** Flag indicating which Microsoft Office products have this setting enabled. The FEATURE\_UNC\_SAVEDFILECHECK feature enables the Mark of the Web (MOTW) for local files loaded from network locations that have been shared by using the Universal Naming Convention (UNC)
- **OleValidateUrl** Flag indicating which Microsoft Office products have this setting enabled. When enabled, the FEATURE\_VALIDATE\_NAVIGATE\_URL feature control prevents Windows Internet Explorer from navigating to a badly formed URL
- **OleWebOcPopup** Flag indicating which Microsoft Office products have this setting enabled. The FEATURE\_WEBOC\_POPUPMANAGEMENT feature allows applications hosting the WebBrowser Control to receive the default Internet Explorer pop-up window management behavior
- **OleWinRestrict** Flag indicating which Microsoft Office products have this setting enabled. When enabled, the FEATURE\_WINDOW\_RESTRICTIONS feature adds several restrictions to the size and behavior of popup windows
- **OleZoneElevate** Flag indicating which Microsoft Office products have this setting enabled. When enabled, the FEATURE\_ZONE\_ELEVATION feature prevents pages in one zone from navigating to pages in a higher security zone unless the navigation is generated by the user

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeIESettingsStartSync**

Diagnostic event to indicate a new sync is being generated for this object type.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeIdentifiersAdd**

This event provides data on the Office identifiers

The following fields are available:

- **OAudienceData** Sub-identifier for Microsoft Office release management, identifying the pilot group for a



device

- **OAudienceId** Microsoft Office identifier for Microsoft Office release management, identifying the pilot group for a device
- **OMID** Identifier for the Office SQM Machine
- **OPlatform** Whether the installed Microsoft Office product is 32-bit or 64-bit
- **OTenantId** Unique GUID representing the Microsoft O365 Tenant
- **OVersion** Installed version of Microsoft Office. For example, 16.0.8602.1000
- **OWowMID** Legacy Microsoft Office telemetry identifier (SQM Machine ID) for WoW systems (32-bit Microsoft Office on 64-bit Windows)

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeIdentifiersStartSync**

Diagnostic event to indicate a new sync is being generated for this object type.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeInsightsAdd**

This event provides insight data on the installed Office products

The following fields are available:

- **OfficeApplication** The name of the Office application.
- **OfficeArchitecture** The bitness of the Office application.
- **OfficeVersion** The version of the Office application.
- **Value** The insights collected about this entity.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeInsightsRemove**

Indicates that this particular data object represented by the objectInstanceId is no longer present.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeInsightsStartSync**

This diagnostic event indicates that a new sync is being generated for this object type.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeProductsAdd**

This event list all installed Office products

The following fields are available:

- **OC2rApps** A GUID that describes the Office Click-To-Run apps
- **OC2rSkus** Comma-delimited list (CSV) of Office Click-To-Run products installed on the device. For example, Office 2016 ProPlus
- **OMsiApps** Comma-delimited list (CSV) of Office MSI products installed on the device. For example, Microsoft Word
- **OProductCodes** A GUID that describes the Office MSI products

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeProductsStartSync**

Diagnostic event to indicate a new sync is being generated for this object type.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeSettingsAdd**

This event describes various Office settings

The following fields are available:

- **BrowserFlags** Browser flags for Office-related products
- **ExchangeProviderFlags** Provider policies for Office Exchange
- **SharedComputerLicensing** Office shared computer licensing policies

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeSettingsStartSync**

Diagnostic event to indicate a new sync is being generated for this object type.

### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeVBAAdd**

This event provides a summary rollup count of conditions encountered while performing a local scan of Office files, analyzing for known VBA programmability compatibility issues between legacy office version and ProPlus, and between 32 and 64-bit versions

The following fields are available:

- **Design** Count of files with design issues found
- **Design\_x64** Count of files with 64 bit design issues found
- **DuplicateVBA** Count of files with duplicate VBA code
- **HasVBA** Count of files with VBA code
- **Inaccessible** Count of files that were inaccessible for scanning
- **Issues** Count of files with issues detected
- **Issues\_x64** Count of files with 64-bit issues detected
- **IssuesNone** Count of files with no issues detected
- **IssuesNone\_x64** Count of files with no 64-bit issues detected
- **Locked** Count of files that were locked, preventing scanning
- **NoVBA** Count of files with no VBA inside
- **Protected** Count of files that were password protected, preventing scanning
- **RemLimited** Count of files that require limited remediation changes
- **RemLimited\_x64** Count of files that require limited remediation changes for 64-bit issues
- **RemSignificant** Count of files that require significant remediation changes
- **RemSignificant\_x64** Count of files that require significant remediation changes for 64-bit issues
- **Score** Overall compatibility score calculated for scanned content
- **Score\_x64** Overall 64-bit compatibility score calculated for scanned content
- **Total** Total number of files scanned
- **Validation** Count of files that require additional manual validation
- **Validation\_x64** Count of files that require additional manual validation for 64-bit issues

### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeVBARemove**

Indicates that this particular data object represented by the objectInstanceId is no longer present.

### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeVBARuleViolationsAdd**

This event provides data on Microsoft Office VBA rule violations, including a rollup count per violation type, giving an indication of remediation requirements for an organization. The event identifier is a unique GUID, associated with the validation rule

The following fields are available:

- **Count** Count of total Microsoft Office VBA rule violations

### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeVBARuleViolationsRemove**

Indicates that this particular data object represented by the objectInstanceId is no longer present.

### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeVBARuleViolationsStartSync**

This event indicates that a new sync is being generated for this object type.

### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeVBAStartSync**

Diagnostic event to indicate a new sync is being generated for this object type.

### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousUUPInfoAdd**

Provides data on Unified Update Platform (UUP) products and what version they are at.

The following fields are available:

- **Identifier** UUP identifier
- **LastActivatedVersion** Last activated version
- **PreviousVersion** Previous version
- **Source** UUP source
- **Version** UUP version

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousUUPInfoRemove**

Indicates that this particular data object represented by the objectInstanceId is no longer present.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousUUPInfoStartSync**

Diagnostic event to indicate a new sync is being generated for this object type.

#### **Microsoft.Windows.Inventory.Indicators.Checksum**

This event summarizes the counts for the InventoryMiscellaneousUexIndicatorAdd events.

The following fields are available:

- **ChecksumDictionary** A count of each operating system indicator.
- **PCFP** Equivalent to the InventoryId field that is found in other core events.

#### **Microsoft.Windows.Inventory.Indicators.InventoryMiscellaneousUexIndicatorAdd**

These events represent the basic metadata about the OS indicators installed on the system which are used for keeping the device up-to-date.

The following fields are available:

- **IndicatorValue** The indicator value

#### **Microsoft.Windows.Inventory.Indicators.InventoryMiscellaneousUexIndicatorRemove**

This event is a counterpart to InventoryMiscellaneousUexIndicatorAdd that indicates that the item has been removed.

#### **Microsoft.Windows.Inventory.Indicators.InventoryMiscellaneousUexIndicatorStartSync**

This event indicates that a new set of InventoryMiscellaneousUexIndicatorAdd events will be sent.

## Microsoft Store events

#### **Microsoft.Windows.StoreAgent.Telemetry.AbortedInstallation**

This event is sent when an installation or update is canceled by a user or the system and is used to help keep Windows Apps up to date and secure.

The following fields are available:

- **AggregatedPackageFullNames** The names of all packages to be downloaded and installed.
- **AttemptNumber** Number of retry attempts before it was canceled.
- **BundleId** The Item Bundle ID.
- **CategoryId** The Item Category ID.
- **ClientAppId** The identity of the app that initiated this operation.
- **HResult** The result code of the last action performed before this operation.
- **IsBundle** Is this a bundle?
- **IsInteractive** Was this requested by a user?
- **IsMandatory** Was this a mandatory update?
- **IsRemediation** Was this a remediation install?

- **IsRestore** Is this automatically restoring a previously acquired product?
- **IsUpdate** Flag indicating if this is an update.
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **PFN** The product family name of the product being installed.
- **ProductId** The identity of the package or packages being installed.
- **SystemAttemptNumber** The total number of automatic attempts at installation before it was canceled.
- **UserAttemptNumber** The total number of user attempts at installation before it was canceled.
- **WUContentId** The Windows Update content ID

#### **Microsoft.Windows.StoreAgent.Telemetry.BeginGetInstalledContentIds**

This event is sent when an inventory of the apps installed is started to determine whether updates for those apps are available. It's used to help keep Windows up-to-date and secure.

#### **Microsoft.Windows.StoreAgent.Telemetry.BeginUpdateMetadataPrepare**

This event is sent when the Store Agent cache is refreshed with any available package updates. It's used to help keep Windows up-to-date and secure.

#### **Microsoft.Windows.StoreAgent.Telemetry.CancelInstallation**

This event is sent when an app update or installation is canceled while in interactive mode. This can be canceled by the user or the system. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **AggregatedPackageFullNames** The names of all package or packages to be downloaded and installed.
- **AttemptNumber** Total number of installation attempts.
- **BundleId** The identity of the Windows Insider build that is associated with this product.
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **IsBundle** Is this a bundle?
- **IsInteractive** Was this requested by a user?
- **IsMandatory** Is this a mandatory update?
- **IsRemediation** Is this repairing a previous installation?
- **IsRestore** Is this an automatic restore of a previously acquired product?
- **IsUpdate** Is this a product update?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **PFN** The name of all packages to be downloaded and installed.
- **PreviousHResult** The previous HResult code.
- **PreviousInstallState** Previous installation state before it was canceled.
- **ProductId** The name of the package or packages requested for installation.
- **RelatedCV** Correlation Vector of a previous performed action on this product.
- **SystemAttemptNumber** Total number of automatic attempts to install before it was canceled.
- **UserAttemptNumber** Total number of user attempts to install before it was canceled.
- **WUContentId** The Windows Update content ID

#### **Microsoft.Windows.StoreAgent.Telemetry.CompleteInstallOperationRequest**

This event is sent after the app installations or updates. It's used to help keep Windows up-to-date and secure

The following fields are available:

- **CatalogId** The Store Product ID of the app being installed.
- **HResult** HResult code of the action being performed.

- **IsBundle** Is this a bundle?
- **PackageFamilyName** The name of the package being installed.
- **ProductId** The Store Product ID of the product being installed.
- **Skuld** Specific edition of the item being installed.

#### **Microsoft.Windows.StoreAgent.Telemetry.EndAcquireLicense**

This event is sent after the license is acquired when a product is being installed. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **AggregatedPackageFullNames** Includes a set of package full names for each app that is part of an atomic set.
- **AttemptNumber** The total number of attempts to acquire this product.
- **BundleId** The bundle ID
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **HResult** HRESULT code to show the result of the operation (success/failure).
- **IsBundle** Is this a bundle?
- **IsInteractive** Did the user initiate the installation?
- **IsMandatory** Is this a mandatory update?
- **IsRemediation** Is this repairing a previous installation?
- **IsRestore** Is this happening after a device restore?
- **IsUpdate** Is this an update?
- **ParentBundleId** The parent bundle ID (if it's part of a bundle).
- **PFN** Product Family Name of the product being installed.
- **ProductId** The Store Product ID for the product being installed.
- **SystemAttemptNumber** The number of attempts by the system to acquire this product.
- **UserAttemptNumber** The number of attempts by the user to acquire this product
- **WUContentId** The Windows Update content ID

#### **Microsoft.Windows.StoreAgent.Telemetry.EndDownload**

This event happens during the app update or installation when content is being downloaded at the end of the process to report success or failure. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **AggregatedPackageFullNames** The name of all packages to be downloaded and installed.
- **AttemptNumber** Number of retry attempts before it was canceled.
- **BundleId** The identity of the Windows Insider build associated with this product.
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **DownloadSize** The total size of the download.
- **ExtendedHResult** Any extended HRESULT error codes.
- **HResult** The result code of the last action performed.
- **IsBundle** Is this a bundle?
- **IsInteractive** Is this initiated by the user?
- **IsMandatory** Is this a mandatory installation?
- **IsRemediation** Is this repairing a previous installation?
- **IsRestore** Is this a restore of a previously acquired product?

- **IsUpdate** Is this an update?
- **ParentBundleId** The parent bundle ID (if it's part of a bundle).
- **PFN** The Product Family Name of the app being download.
- **ProductId** The Store Product ID for the product being installed.
- **SystemAttemptNumber** The number of attempts by the system to download.
- **UserAttemptNumber** The number of attempts by the user to download.
- **WUContentId** The Windows Update content ID.

#### **Microsoft.Windows.StoreAgent.Telemetry.EndFrameworkUpdate**

This event happens when an app update requires an updated Framework package and the process starts to download it. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **HResult** The result code of the last action performed before this operation.

#### **Microsoft.Windows.StoreAgent.Telemetry.EndGetInstalledContentIds**

This event is sent after sending the inventory of the products installed to determine whether updates for those products are available. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **HResult** The result code of the last action performed before this operation.

#### **Microsoft.Windows.StoreAgent.Telemetry.EndInstall**

This event is sent after a product has been installed. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **AggregatedPackageFullNames** The names of all packages to be downloaded and installed.
- **AttemptNumber** The number of retry attempts before it was canceled.
- **BundleId** The identity of the build associated with this product.
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **ExtendedHResult** The extended HResult error code.
- **HResult** The result code of the last action performed.
- **IsBundle** Is this a bundle?
- **IsInteractive** Is this an interactive installation?
- **IsMandatory** Is this a mandatory installation?
- **IsRemediation** Is this repairing a previous installation?
- **IsRestore** Is this automatically restoring a previously acquired product?
- **IsUpdate** Is this an update?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **PFN** Product Family Name of the product being installed.
- **ProductId** The Store Product ID for the product being installed.
- **SystemAttemptNumber** The total number of system attempts.
- **UserAttemptNumber** The total number of user attempts.
- **WUContentId** The Windows Update content ID

#### **Microsoft.Windows.StoreAgent.Telemetry.EndScanForUpdates**

This event is sent after a scan for product updates to determine if there are packages to install. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **ClientAppId** The identity of the app that initiated this operation.
- **HResult** The result code of the last action performed.
- **IsApplicability** Is this request to only check if there are any applicable packages to install?
- **IsInteractive** Is this user requested?
- **IsOnline** Is the request doing an online check?

### **Microsoft.Windows.StoreAgent.Telemetry.EndSearchUpdatePackages**

This event is sent after searching for update packages to install. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **AggregatedPackageFullNames** The names of all packages to be downloaded and installed.
- **AttemptNumber** The total number of retry attempts before it was canceled.
- **BundleId** The identity of the build associated with this product.
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **HResult** The result code of the last action performed.
- **IsBundle** Is this a bundle?
- **IsInteractive** Is this user requested?
- **IsMandatory** Is this a mandatory update?
- **IsRemediation** Is this repairing a previous installation?
- **IsRestore** Is this restoring previously acquired content?
- **IsUpdate** Is this an update?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **PFN** The name of the package or packages requested for install.
- **ProductId** The Store Product ID for the product being installed.
- **SystemAttemptNumber** The total number of system attempts.
- **UserAttemptNumber** The total number of user attempts.
- **WUContentId** The Windows Update content ID

### **Microsoft.Windows.StoreAgent.Telemetry.EndStageUserData**

This event is sent between download and installation to see if there is app data that needs to be restored from the cloud. It's used to keep Windows up-to-date and secure.

The following fields are available:

- **AggregatedPackageFullNames** The name of all packages to be downloaded and installed.
- **AttemptNumber** The total number of retry attempts before it was canceled.
- **BundleId** The identity of the build associated with this product.
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **HResult** The result code of the last action performed.
- **IsBundle** Is this a bundle?
- **IsInteractive** Is this user requested?
- **IsMandatory** Is this a mandatory update?
- **IsRemediation** Is this repairing a previous installation?
- **IsRestore** Is this restoring previously acquired content?
- **IsUpdate** Is this an update?

- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **PFN** The name of the package or packages requested for install.
- **ProductId** The Store Product ID for the product being installed.
- **SystemAttemptNumber** The total number of system attempts.
- **UserAttemptNumber** The total number of system attempts.
- **WUContentId** The Windows Update content ID

#### **Microsoft.Windows.StoreAgent.Telemetry.EndUpdateMetadataPrepare**

This event happens after a scan for available app updates. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **HResult** The result code of the last action performed.

#### **Microsoft.Windows.StoreAgent.Telemetry.FulfillmentComplete**

The FulfillmentComplete event is fired at the end of an app install or update. We use this to track the very end of the install/update process. StoreAgent events are needed to help keep Windows pre-installed 1st party apps up to date and secure, such as the mail and calendar apps. App update failure can be unique across devices and without this data from every device we will not be able to track the success/failure and fix any future vulnerabilities related to these built in Windows Apps.

The following fields are available:

- **CatalogId** The CatalogId is the name of the product catalog from which this app was chosen.
- **FailedRetry** Was the installation or update retry successful?
- **HResult** The HResult code of the operation.
- **PFN** The Package Family Name of the app that is being installed or updated.
- **ProductId** The product ID of the app that is being updated or installed.

#### **Microsoft.Windows.StoreAgent.Telemetry.FulfillmentInitiate**

The FulfillmentInitiate event is fired at the start of an app install or update. We use this to track the very beginning of the install/update process. StoreAgent events are needed to help keep Windows pre-installed 1st party apps up to date and secure, such as the mail and calendar apps. App update failure can be unique across devices and without this data from every device we will not be able to track the success/failure and fix any future vulnerabilities related to these built in Windows Apps.

The following fields are available:

- **PFN** The Package Family Name of the app that is being installed or updated.
- **ProductId** The product ID of the app that is being updated or installed.
- **CatalogId** The CatalogId is the name of the product catalog from which this app was chosen.

#### **Microsoft.Windows.StoreAgent.Telemetry.InstallOperationRequest**

This event happens at the beginning of the install process when an app update or new app is installed. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **BundleId** The identity of the build associated with this product.
- **CatalogId** If this product is from a private catalog, the Store Product ID for the product being installed.
- **ProductId** The Store Product ID for the product being installed.
- **Skuid** Specific edition ID being installed.
- **VolumePath** The disk path of the installation.

#### **Microsoft.Windows.StoreAgent.Telemetry.PauseInstallation**



This event is sent when a product install or update is paused either by a user or the system. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **AggregatedPackageFullNames** The names of all packages to be downloaded and installed.
- **AttemptNumber** The total number of retry attempts before it was canceled.
- **BundleId** The identity of the build associated with this product.
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **IsBundle** Is this a bundle?
- **IsInteractive** Is this user requested?
- **IsMandatory** Is this a mandatory update?
- **IsRemediation** Is this repairing a previous installation?
- **IsRestore** Is this restoring previously acquired content?
- **IsUpdate** Is this an update?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **PFN** The Product Full Name.
- **PreviousHResult** The result code of the last action performed before this operation.
- **PreviousInstallState** Previous state before the installation or update was paused.
- **ProductId** The Store Product ID for the product being installed.
- **RelatedCV** Correlation Vector of a previous performed action on this product.
- **SystemAttemptNumber** The total number of system attempts.
- **UserAttemptNumber** The total number of user attempts.
- **WUContentId** The Windows Update content ID

#### **Microsoft.Windows.StoreAgent.Telemetry.ResumeInstallation**

This event happens when a product install or update is resumed either by a user or the system. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **AggregatedPackageFullNames** The names of all packages to be downloaded and installed.
- **AttemptNumber** The number of retry attempts before it was canceled.
- **BundleId** The identity of the build associated with this product.
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **HResult** The result code of the last action performed before this operation.
- **IsBundle** Is this a bundle?
- **IsInteractive** Is this user requested?
- **IsMandatory** Is this a mandatory update?
- **IsRemediation** Is this repairing a previous installation?
- **IsRestore** Is this restoring previously acquired content?
- **IsUpdate** Is this an update?
- **IsUserRetry** Did the user initiate the retry?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **PFN** The name of the package or packages requested for install.
- **PreviousHResult** The previous HResult error code.
- **PreviousInstallState** Previous state before the installation was paused.
- **ProductId** The Store Product ID for the product being installed.

- **RelatedCV** Correlation Vector for the original install before it was resumed.
- **SystemAttemptNumber** The total number of system attempts.
- **UserAttemptNumber** The total number of user attempts.
- **WUContentId** The Windows Update content ID

#### **Microsoft.Windows.StoreAgent.Telemetry.ResumeOperationRequest**

This event happens when a product install or update is resumed by a user and on installation retries. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **ProductId** The Store Product ID for the product being installed.

#### **Microsoft.Windows.StoreAgent.Telemetry.SearchForUpdateOperationRequest**

This event is sent when searching for update packages to install. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **CatalogId** The Store Product ID for the product being installed.
- **ProductId** The Store Product ID for the product being installed.
- **Skuid** Specific edition of the app being updated.

#### **Microsoft.Windows.StoreAgent.Telemetry.UpdateAppOperationRequest**

This event happens an app for a user needs to be updated. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **PFamN** The name of the product that is requested for update.

## Privacy consent logging events

#### **Microsoft.Windows.Shell.PrivacyConsentLogging.PrivacyConsentCompleted**

This event is used to determine whether the user successfully completed the privacy consent experience.

The following fields are available:

- **presentationVersion** Which display version of the privacy consent experience the user completed
- **privacyConsentState** The current state of the privacy consent experience
- **settingsVersion** Which setting version of the privacy consent experience the user completed
- **userOobeExitReason** The exit reason of the privacy consent experience

#### **Microsoft.Windows.Shell.PrivacyConsentLogging.PrivacyConsentStatus**

Event tells us effectiveness of new privacy experience.

The following fields are available:

- **isAdmin** Whether the current user is an administrator or not
- **isLaunching** Whether or not the privacy consent experience will be launched
- **isSilentElevation** Whether the current user has enabled silent elevation
- **privacyConsentState** The current state of the privacy consent experience
- **userRegionCode** The current user's region setting

## Setup events

#### **SetupPlatformTel.SetupPlatformTelEvent**

This service retrieves events generated by SetupPlatform, the engine that drives the various deployment scenarios.

The following fields are available:

- **FieldName** Retrieves the event name/data point. Examples: InstallStartTime, InstallEndtime, OverallResult etc.
- **GroupName** Retrieves the groupname the event belongs to. Example: Install Information, DU Information, Disk Space Information etc.
- **Value** Retrieves the value associated with the corresponding event name (Field Name). For example: For time related events this will include the system time.

## Shared PC events

### **Microsoft.Windows.SharedPC.AccountManager.DeleteUserAccount**

Activity for deletion of a user account for devices set up for Shared PC mode as part of the Transient Account Manager to help keep Windows up to date. Deleting un-used user accounts on Education/Shared PCs frees up disk space to improve Windows Update success rates.

The following fields are available:

- **accountType** The type of account that was deleted. Example: AD, AAD, or Local
- **deleteState** Whether the attempted deletion of the user account was successful.
- **userSid** The security identifier of the account.
- **wilActivity** Windows Error Reporting data collected when there is a failure in deleting a user account with the Transient Account Manager.

### **Microsoft.Windows.SharedPC.AccountManager.SinglePolicyEvaluation**

Activity for run of the Transient Account Manager that determines if any user accounts should be deleted for devices set up for Shared PC mode to help keep Windows up to date. Deleting unused user accounts on shared devices frees up disk space to improve Windows Update success rates

The following fields are available:

- **totalAccountCount** The number of accounts on a device after running the Transient Account Manager policies.
- **wilActivity** Windows Error Reporting data collected when there is a failure in evaluating accounts to be deleted with the Transient Account Manager.
- **evaluationTrigger** When was the Transient Account Manager policies ran? Example: At log off or during maintenance hours

## SIH events

### **SIHEngineTelemetry.EvalApplicability**

This event is sent when targeting logic is evaluated to determine if a device is eligible for a given action.

The following fields are available:

- **ActionReasons** If an action has been assessed as inapplicable, the additional logic prevented it.
- **AdditionalReasons** If an action has been assessed as inapplicable, the additional logic prevented it.
- **CachedEngineVersion** The engine DLL version that is being used.
- **EventInstanceID** A unique identifier for event instance.
- **EventScenario** Indicates the purpose of sending this event – whether because the software distribution just started checking for content, or whether it was cancelled, succeeded, or failed.
- **HandlerReasons** If an action has been assessed as inapplicable, the installer technology-specific logic prevented it.

- **IsExecutingAction** If the action is presently being executed.
- **ServiceGuid** A unique identifier that represents which service the software distribution client is connecting to (SIH, Windows Update, Windows Store, etc.)
- **SihclientVersion** The client version that is being used.
- **StandardReasons** If an action has been assessed as inapplicable, the standard logic that prevented it.
- **StatusCode** Result code of the event (success, cancellation, failure code HRESULT).
- **UpdateID** A unique identifier for the action being acted upon.
- **WuapiVersion** The Windows Update API version that is currently installed.
- **WuauctVersion** The Windows Update client version that is currently installed.
- **WuauengVersion** The Windows Update engine version that is currently installed.
- **WUDeviceID** The unique identifier controlled by the software distribution client.

### SIHEngineTelemetry.SLSActionData

This event reports if the SIH client was able to successfully parse the manifest describing the actions to be evaluated.

The following fields are available:

- **CachedEngineVersion** The engine DLL version that is being used.
- **EventInstanceID** A unique identifier for event instance.
- **EventScenario** Indicates the purpose of sending this event – whether because the software distribution just started checking for content, or whether it was cancelled, succeeded, or failed.
- **FailedParseActions** The list of actions that were not successfully parsed.
- **ParsedActions** The list of actions that were successfully parsed.
- **ServiceGuid** A unique identifier that represents which service the software distribution client is connecting to (SIH, Windows Update, Windows Store, etc.)
- **SihclientVersion** The client version that is being used.
- **WuapiVersion** The Windows Update API version that is currently installed.
- **WuauctVersion** The Windows Update client version that is currently installed.
- **WuauengVersion** The Windows Update engine version that is currently installed.
- **WUDeviceID** The unique identifier controlled by the software distribution client.

## Software update events

### SoftwareUpdateClientTelemetry.CheckForUpdates

Scan process event on Windows Update client (see eventscenario field for specifics, e.g.: started/failed/succeeded)

The following fields are available:

- **ActivityMatchingId** Contains a unique ID identifying a single CheckForUpdates session from initialization to completion.
- **AllowCachedResults** Indicates if the scan allowed using cached results.
- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client.
- **CurrentMobileOperator** The mobile operator the device is currently connected to.
- **DriverSyncPassPerformed** Were drivers scanned this time?
- **EventScenario** Indicates the purpose of sending this event - whether because the software distribution just started checking for content, or whether it was cancelled, succeeded, or failed.
- **ExtendedStatusCode** Secondary error code for certain scenarios where StatusCode wasn't specific enough.
- **FeatureUpdatePause** Indicates whether feature OS updates are paused on the device.
- **FlightBranch** The branch that a device is on if participating in flighting (pre-release builds).

- **FlightRing** The ring (speed of getting builds) that a device is on if participating in flighting (pre-release builds).
- **HomeMobileOperator** The mobile operator that the device was originally intended to work with.
- **IPVersion** Indicates whether the download took place over IPv4 or IPv6
- **IsWUfBDualScanEnabled** Indicates if Windows Update for Business dual scan is enabled on the device.
- **IsWUfBEnabled** Indicates if Windows Update for Business is enabled on the device.
- **MetadataIntegrityMode** The mode of the update transport metadata integrity check. 0-Unknown, 1-Ignore, 2-Audit, 3-Enforce
- **NumberOfApplicationsCategoryScanEvaluated** The number of categories (apps) for which an app update scan checked
- **NumberOfLoop** The number of round trips the scan required
- **NumberOfNewUpdatesFromServiceSync** The number of updates which were seen for the first time in this scan
- **NumberOfUpdatesEvaluated** The total number of updates which were evaluated as a part of the scan
- **NumFailedMetadataSignatures** The number of metadata signatures checks which failed for new metadata synced down.
- **Online** Indicates if this was an online scan.
- **PhonePreviewEnabled** Indicates whether a phone was getting preview build, prior to flighting (pre-release builds) being introduced.
- **ProcessName** The process name of the caller who initiated API calls, in the event where CallerApplicationName was not provided.
- **QualityUpdatePause** Indicates whether quality OS updates are paused on the device.
- **RelatedCV** The previous Correlation Vector that was used before swapping with a new one
- **ScanDurationInSeconds** The number of seconds a scan took
- **ScanEnqueueTime** The number of seconds it took to initialize a scan
- **ServiceGuid** An ID which represents which service the software distribution client is checking for content (Windows Update, Windows Store, etc.).
- **ServiceUrl** The environment URL a device is configured to scan with
- **ShippingMobileOperator** The mobile operator that a device shipped on.
- **StatusCode** Indicates the result of a CheckForUpdates event (success, cancellation, failure code HRESULT).
- **SyncType** Describes the type of scan the event was
- **TotalNumMetadataSignatures** The total number of metadata signatures checks done for new metadata that was synced down.
- **ApplicableUpdateInfo** Metadata for the updates which were detected as applicable
- **BiosFamily** The family of the BIOS (Basic Input Output System).
- **BiosName** The name of the device BIOS.
- **BiosReleaseDate** The release date of the device BIOS.
- **BiosSKU Number** The sku number of the device BIOS.
- **BIOSVendor** The vendor of the BIOS.
- **BiosVersion** The version of the BIOS.
- **BranchReadinessLevel** The servicing branch configured on the device.
- **ClientVersion** The version number of the software distribution client.
- **DeferralPolicySources** Sources for any update deferral policies defined (GPO = 0x10, MDM = 0x100, Flight = 0x1000, UX = 0x10000).
- **DeferredUpdates** Update IDs which are currently being deferred until a later time
- **DeviceModel** What is the device model.
- **DriverExclusionPolicy** Indicates if the policy for not including drivers with Windows Update is enabled.
- **EventInstanceID** A globally unique identifier for event instance.

- **FeatureUpdateDeferral** The deferral period configured for feature OS updates on the device (in days).
- **FeatureUpdatePausePeriod** The pause duration configured for feature OS updates on the device (in days).
- **IntentPFNs** Intended application-set metadata for atomic update scenarios.
- **NumberOfApplicableUpdates** The number of updates which were ultimately deemed applicable to the system after the detection process is complete
- **PausedUpdates** A list of UpdateIds which that currently being paused.
- **PauseFeatureUpdatesEndTime** If feature OS updates are paused on the device, this is the date and time for the end of the pause time window.
- **PauseFeatureUpdatesStartTime** If feature OS updates are paused on the device, this is the date and time for the beginning of the pause time window.
- **PauseQualityUpdatesEndTime** If quality OS updates are paused on the device, this is the date and time for the end of the pause time window.
- **PauseQualityUpdatesStartTime** If quality OS updates are paused on the device, this is the date and time for the beginning of the pause time window.
- **QualityUpdateDeferral** The deferral period configured for quality OS updates on the device (in days).
- **QualityUpdatePausePeriod** The pause duration configured for quality OS updates on the device (in days).
- **SystemBIOSMajorRelease** Major version of the BIOS.
- **SystemBIOSMinorRelease** Minor version of the BIOS.
- **WebServiceRetryMethods** Web service method requests that needed to be retried to complete operation.
- **WUDeviceID** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue.
- **CachedEngineVersion** For self-initiated healing, the version of the SIH engine that is cached on the device. If the SIH engine does not exist, the value is null.
- **TargetMetadataVersion** For self-initiated healing, this is the target version of the SIH engine to download (if needed). If not, the value is null.
- **IsWUfBFederatedScanDisabled** Indicates if Windows Update for Business federated scan is disabled on the device.
- **CapabilityDetectoidGuid** The GUID for a hardware applicability detectoid that could not be evaluated.
- **CDNCountryCode** Two letter country abbreviation for the CDN's location.
- **CDNId** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue.
- **DriverError** The error code hit during a driver scan. This is 0 if no error was encountered.
- **ExtendedMetadataCabUrl** Hostname that is used to download an update.
- **FailedUpdateGuids** The GUIDs for the updates that failed to be evaluated during the scan.
- **FailedUpdatesCount** The number of updates that failed to be evaluated during the scan.
- **MSIError** The last error that was encountered during a scan for updates.
- **NetworkConnectivityDetected** Indicates the type of network connectivity that was detected. 0 - IPv4, 1 - IPv6
- **Context** Gives context on where the error has occurred. Example: AutoEnable, GetSLSData, AddService, Misc, or Unknown

### **SoftwareUpdateClientTelemetry.Commit**

This event tracks the commit process post the update installation when software update client is trying to update the device.

The following fields are available:

- **BiosFamily** Device family as defined in the system BIOS
- **BiosName** Name of the system BIOS
- **BiosReleaseDate** Release date of the system BIOS

- **BiosSKUNumber** Device SKU as defined in the system BIOS
- **BIOSVendor** Vendor of the system BIOS
- **BiosVersion** Version of the system BIOS
- **BundleId** Identifier associated with the specific content bundle; should not be all zeros if the bundleID was found.
- **BundleRevisionNumber** Identifies the revision number of the content bundle
- **CallerApplicationName** Name provided by the caller who initiated API calls into the software distribution client
- **ClientVersion** Version number of the software distribution client
- **DeviceModel** Device model as defined in the system bios
- **EventInstanceId** A globally unique identifier for event instance
- **EventScenario** Indicates the purpose of the event - whether because scan started, succeeded, failed, etc.
- **EventType** Possible values are "Child", "Bundle", "Release" or "Driver".
- **FlightId** The specific id of the flight the device is getting
- **HandlerType** Indicates the kind of content (app, driver, windows patch, etc.)
- **RevisionNumber** Identifies the revision number of this specific piece of content
- **ServiceGuid** Identifier for the service to which the software distribution client is connecting (Windows Update, Windows Store, etc)
- **SystemBIOSMajorRelease** Major release version of the system bios
- **SystemBIOSMinorRelease** Minor release version of the system bios
- **UpdateId** Identifier associated with the specific piece of content
- **WUDeviceID** Unique device id controlled by the software distribution client

#### SoftwareUpdateClientTelemetry.Download

Download process event for target update on Windows Update client (see eventscenario field for specifics, e.g.: started/failed/succeeded)

The following fields are available:

- **ActiveDownloadTime** How long the download took, in seconds, excluding time where the update wasn't actively being downloaded.
- **AppXBlockHashValidationFailureCount** A count of the number of blocks that have failed validation after being downloaded.
- **AppXDownloadScope** Indicates the scope of the download for application content. For streaming install scenarios, AllContent - non-streaming download, RequiredOnly - streaming download requested content required for launch, AutomaticOnly - streaming download requested automatic streams for the app, and Unknown - for events sent before download scope is determined by the Windows Update client.
- **BiosFamily** The family of the BIOS (Basic Input Output System).
- **BiosName** The name of the device BIOS.
- **BiosReleaseDate** The release date of the device BIOS.
- **BiosSKUNumber** The sku number of the device BIOS.
- **BIOSVendor** The vendor of the BIOS.
- **BiosVersion** The version of the BIOS.
- **BundleBytesDownloaded** How many bytes were downloaded for the specific content bundle.
- **BundleId** Identifier associated with the specific content bundle; should not be all zeros if the bundleID was found.
- **BundleRepeatFailFlag** Indicates whether this particular update bundle had previously failed to download.
- **BundleRevisionNumber** Identifies the revision number of the content bundle.
- **BytesDownloaded** How many bytes were downloaded for an individual piece of content (not the entire

bundle).

- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client.
- **CbsDownloadMethod** Indicates whether the download was a full-file download or a partial/delta download.
- **CDNCountryCode** Two letter country abbreviation for the CDN's location.
- **CDNId** ID which defines which CDN the software distribution client downloaded the content from.
- **ClientVersion** The version number of the software distribution client.
- **CurrentMobileOperator** The mobile operator the device is currently connected to.
- **DeviceModel** What is the device model.
- **DownloadPriority** Indicates whether a download happened at background, normal, or foreground priority.
- **EventInstanceId** A globally unique identifier for event instance.
- **EventScenario** Indicates the purpose of sending this event - whether because the software distribution just started downloading content, or whether it was cancelled, succeeded, or failed.
- **EventType** Possible values are Child, Bundle, or Driver.
- **ExtendedStatusCode** Secondary error code for certain scenarios where StatusCode wasn't specific enough.
- **FeatureUpdatePause** Indicates whether feature OS updates are paused on the device.
- **FlightBranch** The branch that a device is on if participating in flighting (pre-release builds).
- **FlightBuildNumber** If this download was for a flight (pre-release build), this indicates the build number of that flight.
- **FlightId** The specific id of the flight (pre-release build) the device is getting.
- **FlightRing** The ring (speed of getting builds) that a device is on if participating in flighting (pre-release builds).
- **HandlerType** Indicates what kind of content is being downloaded (app, driver, windows patch, etc.).
- **HardwareId** If this download was for a driver targeted to a particular device model, this ID indicates the model of the device.
- **HomeMobileOperator** The mobile operator that the device was originally intended to work with.
- **HostName** The hostname URL the content is downloading from.
- **IPVersion** Indicates whether the download took place over IPv4 or IPv6.
- **IsDependentSet** Indicates whether a driver is a part of a larger System Hardware/Firmware Update
- **IsWUfBDualScanEnabled** Indicates if Windows Update for Business dual scan is enabled on the device.
- **IsWUfBEnabled** Indicates if Windows Update for Business is enabled on the device.
- **NetworkCostBitMask** Indicates what kind of network the device is connected to (roaming, metered, over data cap, etc.)
- **NetworkRestrictionStatus** More general version of NetworkCostBitMask, specifying whether Windows considered the current network to be "metered."
- **PackageFullName** The package name of the content.
- **PhonePreviewEnabled** Indicates whether a phone was opted-in to getting preview builds, prior to flighting (pre-release builds) being introduced.
- **ProcessName** The process name of the caller who initiated API calls, in the event where CallerApplicationName was not provided.
- **QualityUpdatePause** Indicates whether quality OS updates are paused on the device.
- **RegulationReason** The reason that the update is regulated
- **RelatedCV** The previous Correlation Vector that was used before swapping with a new one
- **RepeatFailFlag** Indicates whether this specific piece of content had previously failed to download.
- **RevisionNumber** Identifies the revision number of this specific piece of content.
- **ServiceGuid** An ID which represents which service the software distribution client is installing content for (Windows Update, Windows Store, etc.).
- **Setup360Phase** If the download is for an operating system upgrade, this datapoint indicates which phase of the upgrade is underway.



- **ShippingMobileOperator** The mobile operator that a device shipped on.
- **StatusCode** Indicates the result of a Download event (success, cancellation, failure code HRESULT).
- **SystemBIOSMajorRelease** Major version of the BIOS.
- **SystemBIOSMinorRelease** Minor version of the BIOS.
- **TargetGroupId** For drivers targeted to a specific device model, this ID indicates the distribution group of devices receiving that driver.
- **TargetingVersion** For drivers targeted to a specific device model, this is the version number of the drivers being distributed to the device.
- **ThrottlingServiceHRESULT** Result code (success/failure) while contacting a web service to determine whether this device should download content yet.
- **TimeToEstablishConnection** Time (in ms) it took to establish the connection prior to beginning downloaded.
- **TotalExpectedBytes** The total count of bytes that the download is expected to be.
- **UpdateId** An identifier associated with the specific piece of content.
- **UpdateImportance** Indicates whether a piece of content was marked as Important, Recommended, or Optional.
- **UsedDO** Whether the download used the delivery optimization service.
- **UsedSystemVolume** Indicates whether the content was downloaded to the device's main system storage drive, or an alternate storage drive.
- **WUDeviceID** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue.
- **DownloadScenarioId** A unique ID for a given download used to tie together WU and DO events.

#### SoftwareUpdateClientTelemetry.DownloadCheckpoint

This event provides a checkpoint between each of the Windows Update download phases for UUP content

The following fields are available:

- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client
- **ClientVersion** The version number of the software distribution client
- **EventScenario** Indicates the purpose of sending this event - whether because the software distribution just started checking for content, or whether it was cancelled, succeeded, or failed
- **EventType** Possible values are "Child", "Bundle", "Release" or "Driver"
- **ExtendedStatusCode** Secondary error code for certain scenarios where StatusCode wasn't specific enough
- **FileId** A hash that uniquely identifies a file
- **FileName** Name of the downloaded file
- **FlightId** The unique identifier for each flight
- **RelatedCV** The previous Correlation Vector that was used before swapping with a new one
- **RevisionNumber** Unique revision number of Update
- **ServiceGuid** An ID which represents which service the software distribution client is checking for content (Windows Update, Microsoft Store, etc.)
- **StatusCode** Indicates the result of a CheckForUpdates event (success, cancellation, failure code HRESULT)
- **UpdateId** Unique Update ID
- **WUDeviceID** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue

#### SoftwareUpdateClientTelemetry.DownloadHeartbeat

This event allows tracking of ongoing downloads and contains data to explain the current state of the download

The following fields are available:

- **BytesTotal** Total bytes to transfer for this content
- **BytesTransferred** Total bytes transferred for this content at the time of heartbeat
- **CallerApplicationName** Name provided by the caller who initiated API calls into the software distribution client
- **ClientVersion** The version number of the software distribution client
- **ConnectionStatus** Indicates the connectivity state of the device at the time of heartbeat
- **CurrentError** Last (transient) error encountered by the active download
- **DownloadFlags** Flags indicating if power state is ignored
- **DownloadState** Current state of the active download for this content (queued, suspended, or progressing)
- **EventType** Possible values are "Child", "Bundle", or "Driver"
- **FlightId** The unique identifier for each flight
- **IsNetworkMetered** Indicates whether Windows considered the current network to be "metered"
- **MOAppDownloadLimit** Mobile operator cap on size of application downloads, if any
- **MOUpdateDownloadLimit** Mobile operator cap on size of operating system update downloads, if any
- **PowerState** Indicates the power state of the device at the time of heartbeat (DC, AC, Battery Saver, or Connected Standby)
- **RelatedCV** The previous correlation vector that was used by the client, before swapping with a new one
- **ResumeCount** Number of times this active download has resumed from a suspended state
- **RevisionNumber** Identifies the revision number of this specific piece of content
- **ServiceGuid** Identifier for the service to which the software distribution client is connecting (Windows Update, Microsoft Store, etc)
- **SuspendCount** Number of times this active download has entered a suspended state
- **SuspendReason** Last reason for why this active download entered a suspended state
- **UpdateId** Identifier associated with the specific piece of content
- **WUDeviceID** Unique device id controlled by the software distribution client

### SoftwareUpdateClientTelemetry.Install

This event sends tracking data about the software distribution client installation of the content for that update, to help keep Windows up to date.

The following fields are available:

- **BiosFamily** The family of the BIOS (Basic Input Output System).
- **BiosName** The name of the device BIOS.
- **BiosReleaseDate** The release date of the device BIOS.
- **BiosSKUNumber** The sku number of the device BIOS.
- **BIOSVendor** The vendor of the BIOS.
- **BiosVersion** The version of the BIOS.
- **BundleId** Identifier associated with the specific content bundle; should not be all zeros if the bundleID was found.
- **BundleRepeatFailFlag** Has this particular update bundle previously failed to install?
- **BundleRevisionNumber** Identifies the revision number of the content bundle.
- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client.
- **ClientVersion** The version number of the software distribution client.
- **CSLErrorType** The stage of CBS installation where it failed.
- **CurrentMobileOperator** Mobile operator that device is currently connected to.
- **DeviceModel** What is the device model.
- **DriverPingBack** Contains information about the previous driver and system state.

- **EventInstanceId** A globally unique identifier for event instance.
- **EventScenario** Indicates the purpose of sending this event - whether because the software distribution just started installing content, or whether it was cancelled, succeeded, or failed.
- **EventType** Possible values are Child, Bundle, or Driver.
- **ExtendedErrorCode** The extended error code.
- **ExtendedStatusCode** Secondary error code for certain scenarios where StatusCode wasn't specific enough.
- **FeatureUpdatePause** Are feature OS updates paused on the device?
- **FlightBranch** The branch that a device is on if participating in the Windows Insider Program.
- **FlightBuildNumber** If this installation was for a Windows Insider build, this is the build number of that build.
- **FlightId** The specific ID of the Windows Insider build the device is getting.
- **FlightRing** The ring that a device is on if participating in the Windows Insider Program.
- **HandlerType** Indicates what kind of content is being installed. Example: app, driver, Windows update
- **HardwareId** If this install was for a driver targeted to a particular device model, this ID indicates the model of the device.
- **HomeMobileOperator** The mobile operator that the device was originally intended to work with.
- **IntentPFNs** Intended application-set metadata for atomic update scenarios.
- **IsDependentSet** Is the driver part of a larger System Hardware/Firmware update?
- **IsFinalOutcomeEvent** Does this event signal the end of the update/upgrade process?
- **IsFirmware** Is this update a firmware update?
- **IsSuccessFailurePostReboot** Did it succeed and then fail after a restart?
- **IsWUfBDualScanEnabled** Is Windows Update for Business dual scan enabled on the device?
- **IsWUfBEnabled** Is Windows Update for Business enabled on the device?
- **MergedUpdate** Was the OS update and a BSP update merged for installation?
- **MsiAction** The stage of MSI installation where it failed.
- **MsiProductCode** The unique identifier of the MSI installer.
- **PackageFullName** The package name of the content being installed.
- **PhonePreviewEnabled** Indicates whether a phone was getting preview build, prior to flighting being introduced.
- **ProcessName** The process name of the caller who initiated API calls, in the event where CallerApplicationName was not provided.
- **QualityUpdatePause** Are quality OS updates paused on the device?
- **RelatedCV** The previous Correlation Vector that was used before swapping with a new one
- **RepeatFailFlag** Indicates whether this specific piece of content had previously failed to install.
- **RevisionNumber** The revision number of this specific piece of content.
- **ServiceGuid** An ID which represents which service the software distribution client is installing content for (Windows Update, Windows Store, etc.).
- **Setup360Phase** If the install is for an operating system upgrade, indicates which phase of the upgrade is underway.
- **ShippingMobileOperator** The mobile operator that a device shipped on.
- **StatusCode** Indicates the result of an installation event (success, cancellation, failure code HRESULT).
- **SystemBIOSMajorRelease** Major version of the BIOS.
- **SystemBIOSMinorRelease** Minor version of the BIOS.
- **TargetGroupId** For drivers targeted to a specific device model, this ID indicates the distribution group of devices receiving that driver.
- **TargetingVersion** For drivers targeted to a specific device model, this is the version number of the drivers being distributed to the device.
- **TransactionCode** The ID which represents a given MSI installation

- **UpdateId** Unique update ID
- **UpdateImportance** Indicates whether a piece of content was marked as Important, Recommended, or Optional.
- **UsedSystemVolume** Indicates whether the content was downloaded and then installed from the device's main system storage drive, or an alternate storage drive.
- **WUDeviceID** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue.

### SoftwareUpdateClientTelemetry.UpdateDetected

This event sends data about an AppX app that has been updated from the Microsoft Store, including what app needs an update and what version/architecture is required, in order to understand and address problems with apps getting required updates.

The following fields are available:

- **ApplicableUpdateInfo** Metadata for the updates which were detected as applicable
- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client
- **IntentPFNs** Intended application-set metadata for atomic update scenarios.
- **NumberOfApplicableUpdates** The number of updates which were ultimately deemed applicable to the system after the detection process is complete
- **RelatedCV** The previous Correlation Vector that was used before swapping with a new one
- **ServiceGuid** An ID which represents which service the software distribution client is connecting to (Windows Update, Windows Store, etc.)
- **WUDeviceID** The unique device ID controlled by the software distribution client

### SoftwareUpdateClientTelemetry.UpdateMetadataIntegrity

Ensures Windows Updates are secure and complete. Event helps to identify whether update content has been tampered with and protects against man-in-the-middle attack.

The following fields are available:

- **EndpointUrl** URL of the endpoint where client obtains update metadata. Used to identify test vs staging vs production environments.
- **EventScenario** Indicates the purpose of the event - whether because scan started, succeeded, failed, etc.
- **ExtendedStatusCode** Secondary status code for certain scenarios where StatusCode was not specific enough.
- **LeafCertId** Integral id from the FragmentSigning data for certificate which failed.
- **MetadataIntegrityMode** Mode of update transport metadata integrity check. 0-Unknown, 1-Ignoe, 2-Audit, 3-Enforce
- **MetadataSignature** Base64 string of the signature associated with the update metadata (specified by revision id)
- **RevisionId** Identifies the revision of this specific piece of content
- **RevisionNumber** Identifies the revision number of this specific piece of content
- **ServiceGuid** Identifier for the service to which the software distribution client is connecting (Windows Update, Windows Store, etc)
- **SHA256OfLeafCertPublicKey** Base64 encoding of hash of the Base64CertData in the FragmentSigning data of leaf certificate.
- **SHA256OfTimestampToken** Base64 string of hash of the timestamp token blob
- **SignatureAlgorithm** Hash algorithm for the metadata signature
- **SLSPrograms** A test program a machine may be opted in. Examples include "Canary" and "Insider Fast".
- **StatusCode** Result code of the event (success, cancellation, failure code HRESULT)

- **TimestampTokenId** Created time encoded in the timestamp blob. This will be zeroed if the token is itself malformed and decoding failed.
- **UpdateId** Identifier associated with the specific piece of content
- **RawMode** Raw unparsed mode string from the SLS response. May be null if not applicable.
- **TimestampTokenCertThumbprint** The thumbprint of the encoded timestamp token.
- **ValidityWindowInDays** The validity window that's in effect when verifying the timestamp.
- **CallerApplicationName** Name of application making the Windows Update request. Used to identify context of request.
- **ListOfSHA256OfIntermediateCerData** A semicolon delimited list of base64 encoding of hashes for the Base64CerData in the FragmentSigning data of an intermediate certificate.
- **RawValidityWindowInDays** The raw unparsed validity window string in days of the timestamp token. This field is null if not applicable.
- **SHA256OfLeafCerData** A base64 encoding of the hash for the Base64CerData in the FragmentSigning data of the leaf certificate.

## Update events

### Update360Telemetry.UpdateAgentCommit

This event collects information regarding the commit phase of the new UUP (Unified Update Platform) update scenario, which is leveraged by both Mobile and Desktop.

The following fields are available:

- **ErrorCode** The error code returned for the current install phase.
- **FlightId** Unique ID for each flight.
- **ObjectId** Unique value for each Update Agent mode.
- **RelatedCV** Correlation vector value generated from the latest USO scan.
- **Result** Outcome of the install phase of the update.
- **ScenarioId** Indicates the update scenario.
- **SessionId** Unique value for each update attempt.
- **UpdateId** Unique ID for each update.

### Update360Telemetry.UpdateAgentDownloadRequest

The UpdateAgent\_DownloadRequest event sends data for the download request phase of updating Windows via the new UUP (Unified Update Platform) scenario. Applicable to PC and Mobile.

The following fields are available:

- **DeletedCorruptFiles** Boolean indicating whether corrupt payload was deleted.
- **ErrorCode** The error code returned for the current download request phase.
- **FlightId** Unique ID for each flight.
- **ObjectId** Unique value for each Update Agent mode (same concept as InstanceId for Setup360)
- **PackageCountOptional** Number of optional packages requested.
- **PackageCountRequired** Number of required packages requested.
- **PackageCountTotal** Total number of packages needed.
- **PackageCountTotalCanonical** Total number of canonical packages.
- **PackageCountTotalDiff** Total number of diff packages.
- **PackageCountTotalExpress** Total number of express packages.
- **PackageSizeCanonical** Size of canonical packages in bytes.
- **PackageSizeDiff** Size of diff packages in bytes.
- **PackageSizeExpress** Size of express packages in bytes.

- **RangeRequestState** Indicates the range request type used.
- **RelatedCV** Correlation vector value generated from the latest USO scan.
- **Result** Outcome of the download request phase of update.
- **ScenarioId** Indicates the update scenario.
- **SessionId** Unique value for each attempt (same value for initialize, download, install commit phases)
- **UpdateId** Unique ID for each update.
- **PackageExpressType** Type of express package.

#### **Update360Telemetry.UpdateAgentExpand**

This event collects information regarding the expansion phase of the new UUP (Unified Update Platform) update scenario; which is leveraged by both Mobile and Desktop.

The following fields are available:

- **ElapsedTickCount** Time taken for expand phase.
- **EndFreeSpace** Free space after expand phase.
- **EndSandboxSize** Sandbox size after expand phase.
- **ErrorCode** The error code returned for the current install phase.
- **FlightId** Unique ID for each flight.
- **ObjectId** Unique value for each Update Agent mode.
- **RelatedCV** Correlation vector value generated from the latest USO scan.
- **ScenarioId** Indicates the update scenario.
- **SessionId** Unique value for each update attempt.
- **StartFreeSpace** Free space before expand phase.
- **StartSandboxSize** Sandbox size after expand phase.
- **UpdateId** Unique ID for each update.

#### **Update360Telemetry.UpdateAgentFellBackToCanonical**

This event collects information when express could not be used and we fall back to canonical during the new UUP (Unified Update Platform) update scenario, which is leveraged by both Mobile and Desktop.

The following fields are available:

- **FlightId** Unique ID for each flight.
- **ObjectId** Unique value for each Update Agent mode.
- **PackageCount** Number of packages that fell back to canonical.
- **PackageList** PackageIds which fell back to canonical.
- **RelatedCV** Correlation vector value generated from the latest USO scan.
- **ScenarioId** Indicates the update scenario.
- **SessionId** Unique value for each update attempt.
- **UpdateId** Unique ID for each update.

#### **Update360Telemetry.UpdateAgentInitialize**

The UpdateAgentInitialize event sends data for the initialize phase of updating Windows via the new UUP (Unified Update Platform) scenario. Applicable to both PCs and Mobile.

The following fields are available:

- **ErrorCode** The error code returned for the current install phase.
- **FlightId** Unique ID for each flight.
- **FlightMetadata** Contains the FlightId and the build being flighted.
- **ObjectId** Unique value for each Update Agent mode.

- **RelatedCV** Correlation vector value generated from the latest USO scan.
- **Result** Outcome of the install phase of the update.
- **ScenarioId** Indicates the update scenario.
- **SessionData** String containing instructions to update agent for processing FODs and DUICs (Null for other scenarios).
- **SessionId** Unique value for each update attempt.
- **UpdateId** Unique ID for each update.

#### **Update360Telemetry.UpdateAgentInstall**

The UpdateAgentInstall event sends data for the install phase of updating Windows.

The following fields are available:

- **ErrorCode** The error code returned for the current install phase.
- **FlightId** Unique value for each Update Agent mode (same concept as InstanceId for Setup360).
- **ObjectId** Correlation vector value generated from the latest USO scan.
- **RelatedCV** Correlation vector value generated from the latest USO scan.
- **Result** The result for the current install phase.
- **ScenarioId** Indicates the update scenario.
- **SessionId** Unique value for each update attempt.
- **UpdateId** Unique ID for each update.

#### **Update360Telemetry.UpdateAgentMerge**

The UpdateAgentMerge event sends data on the merge phase when updating Windows.

The following fields are available:

- **ErrorCode** The error code returned for the current merge phase.
- **FlightId** Unique ID for each flight.
- **ObjectId** Unique value for each Update Agent mode.
- **RelatedCV** Related correlation vector value.
- **Result** Outcome of the merge phase of the update.
- **ScenarioId** Indicates the update scenario.
- **SessionId** Unique value for each attempt.
- **UpdateId** Unique ID for each update.

#### **Update360Telemetry.UpdateAgentModeStart**

The UpdateAgentModeStart event sends data for the start of each mode during the process of updating Windows via the new UUP (Unified Update Platform) scenario. Applicable to both PCs and Mobile.

The following fields are available:

- **FlightId** Unique ID for each flight.
- **Mode** Indicates the mode that has started.
- **ObjectId** Unique value for each Update Agent mode.
- **RelatedCV** Correlation vector value generated from the latest USO scan.
- **ScenarioId** Indicates the update scenario.
- **SessionId** Unique value for each update attempt.
- **UpdateId** Unique ID for each update.
- **Version** Version of update

#### **Update360Telemetry.UpdateAgentPostRebootResult**

This event collects information for both Mobile and Desktop regarding the post reboot phase of the new UUP (Unified Update Platform) update scenario

The following fields are available:

- **ErrorCode** The error code returned for the current post reboot phase
- **FlightId** The unique identifier for each flight
- **ObjectId** Unique value for each Update Agent mode
- **PostRebootResult** Indicates the Hresult
- **RelatedCV** Correlation vector value generated from the latest USO scan
- **ScenarioId** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **SessionId** Unique value for each update attempt.
- **UpdateId** Unique ID for each update

### **Update360Telemetry.UpdateAgentSetupBoxLaunch**

The UpdateAgent\_SetupBoxLaunch event sends data for the launching of the setup box when updating Windows via the new UUP (Unified Update Platform) scenario. This event is only applicable to PCs.

The following fields are available:

- **FlightId** Unique ID for each flight.
- **FreeSpace** Free space on OS partition.
- **InstallCount** Number of install attempts using the same sandbox.
- **ObjectId** Unique value for each Update Agent mode.
- **Quiet** Indicates whether setup is running in quiet mode.
- **RelatedCV** Correlation vector value generated from the latest USO scan.
- **SandboxSize** Size of the sandbox.
- **ScenarioId** Indicates the update scenario.
- **SessionId** Unique value for each update attempt.
- **SetupMode** Mode of setup to be launched.
- **UpdateId** Unique ID for each Update.
- **UserSession** Indicates whether install was invoked by user actions.
- **ContainsExpressPackage** Indicates whether the download package is express.

## Update notification events

### **Microsoft.Windows.UpdateNotificationPipeline.JavascriptJavascriptCriticalGenericMessage**

Event to indicate that Javascript is reporting a schema and a set of values for critical telemetry.

The following fields are available:

- **CampaignConfigVersion** Config version of current campaign
- **CampaignID** Currently running campaign on UNP
- **ConfigCatalogVersion** Current catalog version of UNP
- **ContentVersion** Content version of the current campaign on UNP
- **CV** Correlation vector
- **DetectorVersion** Most recently run detector version for the current campaign on UNP
- **GlobalEventCounter** Client side counter which indicates ordering of events sent by this user
- **key1** UI interaction data
- **key10** UI interaction data



- **key11** UI interaction data
- **key12** UI interaction data
- **key13** UI interaction data
- **key14** UI interaction data
- **key15** UI interaction data
- **key16** UI interaction data
- **key17** UI interaction data
- **key2** UI interaction data
- **key3** UI interaction data
- **key4** UI interaction data
- **key5** UI interaction data
- **key6** UI interaction data
- **key7** Interaction data for the UI
- **key8** Interaction data for the UI
- **key9** UI interaction data
- **PackageVersion** Current package version of UNP
- **schema** UI interaction type
- **key18** UI interaction data
- **key19** UI interaction data
- **key20** UI interaction data
- **key21** Interaction data for the UI
- **key22** UI interaction data
- **key23** UI interaction data
- **key24** UI interaction data
- **key25** UI interaction data
- **key26** UI interaction data
- **key27** UI interaction data
- **key28** Interaction data for the UI
- **key29** UI interaction data
- **key30** UI interaction data

#### **Microsoft.Windows.UpdateNotificationPipeline.UNPCampaignHeartbeat**

This event is sent at the start of each campaign, to be used as a heartbeat

The following fields are available:

- **CampaignConfigVersion** Configuration version for the current campaign
- **CampaignID** Currently campaign that's running on UNP
- **ConfigCatalogVersion** Current catalog version of UNP
- **ContentVersion** Content version for the current campaign on UNP
- **CV** Correlation vector
- **DetectorVersion** Most recently run detector version for the current campaign on UNP
- **GlobalEventCounter** Client-side counter that indicates the event ordering sent by the user
- **PackageVersion** Current UNP package version

#### **Microsoft.Windows.UpdateNotificationPipeline.UNPCampaignManagerCleaningCampaign**

This event indicates that the Campaign Manager is cleaning up the campaign content

The following fields are available:

- **CampaignConfigVersion** Configuration version for the current campaign
- **CampaignID** Current campaign that's running on UNP
- **ConfigCatalogVersion** Current catalog version of UNP
- **ContentVersion** Content version for the current campaign on UNP
- **CV** Correlation vector
- **DetectorVersion** Most recently run detector version for the current campaign on UNP
- **GlobalEventCounter** Client-side counter that indicates the event ordering sent by the user
- **PackageVersion** Current UNP package version

#### **Microsoft.Windows.UpdateNotificationPipeline.UNPCampaignManagerHeartbeat**

This event is sent at the start of the CampaignManager event and is intended to be used as a heartbeat

The following fields are available:

- **CampaignConfigVersion** Configuration version for the current campaign
- **CampaignID** Currently campaign that's running on UNP
- **ConfigCatalogVersion** Current catalog version of UNP
- **ContentVersion** Content version for the current campaign on UNP
- **CV** Correlation vector
- **DetectorVersion** Most recently run detector version for the current campaign on UNP
- **GlobalEventCounter** Client-side counter that indicates the event ordering sent by the user
- **PackageVersion** Current UNP package version

#### **Microsoft.Windows.UpdateNotificationPipeline.UnpCampaignManagerGetIsCampaignCompleteFailed**

This event is sent when a campaign completion status query fails

The following fields are available:

- **CampaignConfigVersion** Configuration version for the current campaign
- **CampaignID** Current campaign that's running on UNP
- **ConfigCatalogVersion** Current catalog version of UNP
- **ContentVersion** Content version for the current campaign on UNP
- **CV** Correlation vector
- **DetectorVersion** Most recently run detector version for the current campaign on UNP
- **GlobalEventCounter** Client-side counter that indicates the event ordering sent by the user
- **hresult** HRESULT of the failure
- **PackageVersion** Current UNP package version

#### **Microsoft.Windows.UpdateNotificationPipeline.UnpCampaignManagerRunCampaignFailed**

This event is sent when the Campaign Manager encounters an unexpected error while running the campaign

The following fields are available:

- **CampaignConfigVersion** Configuration version for the current campaign
- **CampaignID** Currently campaign that's running on UNP
- **ConfigCatalogVersion** Current catalog version of UNP
- **ContentVersion** Content version for the current campaign on UNP
- **CV** Correlation vector
- **DetectorVersion** Most recently run detector version for the current campaign on UNP
- **GlobalEventCounter** Client-side counter that indicates the event ordering sent by the user
- **hresult** HRESULT of the failure
- **PackageVersion** Current UNP package version

# Upgrade events

## Setup360Telemetry.Downlevel

This event sends data indicating that the device has invoked the downlevel phase of the upgrade. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **ClientId** If using Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, the default value is Media360, but it can be overwritten by the caller to a unique value.
- **HostOSBuildNumber** The build number of the downlevel OS.
- **HostOsSkuName** The operating system edition which is running Setup360 instance (downlevel OS).
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe.
- **ReportId** In the Windows Update scenario, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. It's an HRESULT error code that can be used to diagnose errors.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **SetupVersionBuildNumber** The build number of Setup360 (build number of the target OS).
- **State** Exit state of given Setup360 run. Example: succeeded, failed, blocked, cancelled
- **TestId** A string that uniquely identifies a group of events.
- **Wuld** This is the Windows Update Client ID. In the Windows Update scenario, this is the same as the clientId.
- **FlightData** Unique value that identifies the flight.

## Setup360Telemetry.Finalize

This event sends data indicating that the device has invoked the finalize phase of the upgrade, to help keep Windows up-to-date.

The following fields are available:

- **ClientId** With Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **FlightData** Unique value that identifies the flight.
- **HostOSBuildNumber** The build number of the previous OS.
- **HostOsSkuName** The OS edition which is running Setup360 instance (previous OS).
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe
- **ReportId** With Windows Update, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that is used to diagnose errors.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **TestId** A string to uniquely identify a group of events.
- **Wuld** This is the Windows Update Client ID. With Windows Update, this is the same as the clientId.

## Setup360Telemetry.OsUninstall

The event sends data regarding OS updates and upgrades from Windows 7, Windows 8, and Windows 10. Specifically, the Setup360Telemetry.OSUninstall indicates the outcome of an OS uninstall.

The following fields are available:

- **ClientId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **HostOSBuildNumber** The build number of the previous OS.
- **HostOsSkuName** The OS edition which is running the Setup360 instance (previous OS).
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe.
- **ReportId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that is used to diagnose errors.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).
- **State** Exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **TestId** A string to uniquely identify a group of events.
- **Wuld** Windows Update client ID.
- **FlightData** Unique value that identifies the flight.

#### **Setup360Telemetry.PostRebootInstall**

This event sends data indicating that the device has invoked the postrebootinstall phase of the upgrade, to help keep Windows up-to-date.

The following fields are available:

- **ClientId** With Windows Update, this is the Windows Update client ID that is passed to Setup. In Media setup, the default value is Media360, but can be overwritten by the caller to a unique value.
- **HostOSBuildNumber** The build number of the previous OS.
- **HostOsSkuName** The OS edition which is running Setup360 instance (previous OS).
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe.
- **ReportId** With Windows Update, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that's used to diagnose errors.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **TestId** A string to uniquely identify a group of events.
- **Wuld** This is the Windows Update Client ID. With Windows Update, this is the same as ClientId.
- **FlightData** Unique value that identifies the flight.

#### **Setup360Telemetry.PreDownloadQuiet**

This event sends data indicating that the device has invoked the predownload quiet phase of the upgrade, to help keep Windows up to date.

The following fields are available:

- **ClientId** Using Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **FlightData** Unique value that identifies the flight.
- **HostOSBuildNumber** The build number of the previous OS.
- **HostOsSkuName** The OS edition which is running Setup360 instance (previous operating system).
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe
- **ReportId** Using Windows Update, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that is used to diagnose errors.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, canceled
- **TestId** A string to uniquely identify a group of events.
- **Wuld** This is the Windows Update Client ID. Using Windows Update, this is the same as the clientId.

#### Setup360Telemetry.PreDownloadUX

This event sends data regarding OS Updates and Upgrades from Windows 7.X, Windows 8.X, Windows 10 and RS. Specifically the Setup360Telemetry.PredownloadUX indicates the outcome of the PredownloadUX portion of the update process

The following fields are available:

- **ClientId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **HostOSBuildNumber** The build number of the previous operating system.
- **HostOsSkuName** The OS edition which is running the Setup360 instance (previous operating system).
- **InstanceId** Unique GUID that identifies each instance of setuphost.exe.
- **ReportId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that can be used to diagnose errors.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **SetupVersionBuildNumber** The build number of Setup360 (build number of the target OS).
- **State** The exit state of the Setup360 run. Example: succeeded, failed, blocked, cancelled
- **TestId** A string to uniquely identify a group of events.
- **Wuld** Windows Update client ID.
- **FlightData** In the WU scenario, this will be the WU client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.

#### Setup360Telemetry.PreInstallQuiet

This event sends data indicating that the device has invoked the preinstall quiet phase of the upgrade, to help keep Windows up to date.

The following fields are available:

- **ClientId** With Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **HostOSBuildNumber** The build number of the previous OS.
- **HostOsSkuName** The OS edition which is running Setup360 instance (previous OS).
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe
- **ReportId** With Windows Update, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback etc.
- **Setup360Result** The result of Setup360. This is an HRESULT error code that can be used to diagnose errors.
- **Setup360Scenario** Setup360 flow type (Boot, Media, Update, MCT)
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **TestId** A string to uniquely identify a group of events.
- **Wuld** This is the Windows Update Client ID. With Windows Update, this is the same as the clientId.
- **FlightData** Unique value that identifies the flight.

### Setup360Telemetry.PreInstallUX

This event sends data regarding OS updates and upgrades from Windows 7, Windows 8, and Windows 10. Specifically, the Setup360Telemetry.PreinstallUX indicates the outcome of the PreinstallUX portion of the update process.

The following fields are available:

- **ClientId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **HostOSBuildNumber** The build number of the previous OS.
- **HostOsSkuName** The OS edition which is running the Setup360 instance (previous OS).
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe.
- **ReportId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that is used to diagnose errors.
- **Setup360Scenario** The Setup360 flow type, Example: Boot, Media, Update, MCT
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **TestId** A string to uniquely identify a group of events.
- **Wuld** Windows Update client ID.
- **FlightData** Unique value that identifies the flight.

### Setup360Telemetry.Setup360

This event sends data about OS deployment scenarios, to help keep Windows up-to-date.

The following fields are available:

- **FieldName** Retrieves the data point.
- **FlightData** Specifies a unique identifier for each group of Windows Insider builds.
- **InstanceId** Retrieves a unique identifier for each instance of a setup session.

- **ReportId** Retrieves the report ID.
- **ScenarioId** Retrieves the deployment scenario.
- **Value** Retrieves the value associated with the corresponding FieldName.
- **ClientId** Retrieves the upgrade ID: Upgrades via Windows Update - specifies the WU clientID. All other deployment - static string.

### Setup360Telemetry.UnexpectedEvent

This event sends data indicating that the device has invoked the unexpected event phase of the upgrade, to help keep Windows up to date.

The following fields are available:

- **ClientId** With Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **HostOSBuildNumber** The build number of the previous OS.
- **HostOsSkuName** The OS edition which is running Setup360 instance (previous OS).
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe
- **ReportId** With Windows Update, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that can be used to diagnose errors.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **TestId** A string to uniquely identify a group of events.
- **Wuld** This is the Windows Update Client ID. With Windows Update, this is the same as the clientId.
- **FlightData** Unique value that identifies the flight.

## Windows as a Service diagnostic events

### Microsoft.Windows.WaaSMedic.SummaryEvent

Result of the WaaSMedic operation.

The following fields are available:

- **detectionSummary** Result of each applicable detection that was ran.
- **featureAssessmentImpact** WaaS Assessment impact for feature updates.
- **hrEngineResult** Error code from the engine operation.
- **isManaged** Device is managed for updates.
- **isWUConnected** Device is connected to Windows Update.
- **noMoreActions** No more applicable diagnostics.
- **qualityAssessmentImpact** WaaS Assessment impact for quality updates.
- **remediationSummary** Result of each applicable resolution that was ran.
- **usingBackupFeatureAssessment** Relying on backup feature assessment.
- **usingBackupQualityAssessment** Relying on backup quality assessment.
- **versionString** Version of the WaaSMedic engine.
- **usingCachedFeatureAssessment** WaaS Medic run did not get OS build age from the network on the previous run.
- **usingCachedQualityAssessment** WaaS Medic run did not get OS revision age from the network on the

previous run.

- **insufficientSessions** Device not eligible for diagnostics.

## Windows Error Reporting events

### Windows Error Reporting MTT events

#### **Microsoft.Windows.WER.MTT.Denominator**

This event provides a denominator to calculate MTTF (mean-time-to-failure) for crashes and other errors to help keep Windows up to date.

The following fields are available:

- **Value** Standard UTC emitted DP value structure

### Windows Update CSP events

#### **Microsoft.Windows.UpdateCsp.ExecuteRollBackFeatureFailed**

The Execute Rollback Feature Failed event sends basic telemetry on the failure of the Feature Rollback. This functionality supports our feature by providing IT Admins the ability to see the operation failed, allowing them to do further triage of the device.

The following fields are available:

- **current** Result of currency check
- **dismOperationSucceeded** Dism uninstall operation status
- **hResult** Failure Error code
- **oSVersion** Build number of the machine
- **paused** Machine's pause status
- **rebootRequestSucceeded** Reboot CSP call success status
- **wUfBConnected** Result of WUfB connection check

#### **Microsoft.Windows.UpdateCsp.ExecuteRollBackFeatureNotApplicable**

The Execute Rollback Feature Not Applicable event sends basic telemetry on the applicability of the Feature Rollback, to support the functionality of Feature Rollback. This event provides critical information for the feature because it will alert IT Admins that devices they are attempting to rollback Features updates are not applicable.

The following fields are available:

- **current** Result of currency check
- **dismOperationSucceeded** Dism uninstall operation status
- **oSVersion** Build number of the machine
- **paused** Machine's pause status
- **rebootRequestSucceeded** Reboot CSP call success status
- **wUfBConnected** Result of WUfB connection check

#### **Microsoft.Windows.UpdateCsp.ExecuteRollBackFeatureStarted**

The Execute Rollback Feature Started event sends basic information on the start process to provide information that the Feature Rollback has started.

#### **Microsoft.Windows.UpdateCsp.ExecuteRollBackFeatureSucceeded**

The Execute Rollback Feature Succeed event sends basic telemetry on the success of the Rollback of the Feature updates. This functionality supports our feature by providing insights to IT Admins of the success of the Feature



rollback.

### **Microsoft.Windows.UpdateCsp.ExecuteRollBackQualityFailed**

The Execute Rollback Quality Failed event sends basic telemetry on the failure of the rollback of the Quality/LCU builds. This functionality supports our feature by providing IT Admins the ability to see the operation failed allowing them to do further triage of the device.

The following fields are available:

- **current** Result of currency check
- **dismOperationSucceeded** Dism uninstall operation status
- **hResult** Failure Error code
- **oSVersion** Build number of the machine
- **paused** Machine's pause status
- **rebootRequestSucceeded** Reboot CSP call success status
- **wUfBConnected** Result of WUfB connection check

### **Microsoft.Windows.UpdateCsp.ExecuteRollBackQualityNotApplicable**

The Execute Rollback Quality Not Applicable event sends basic telemetry on the applicability of the Quality Rollback, to support the functionality of Quality Rollback. This event provides critical information for feature because it will alert IT Admins that devices they are attempting to rollback Quality updates are not applicable.

The following fields are available:

- **current** Result of currency check
- **dismOperationSucceeded** Dism uninstall operation status
- **oSVersion** Build number of the machine
- **paused** Machine's pause status
- **rebootRequestSucceeded** Reboot CSP call success status
- **wUfBConnected** Result of WUfB connection check

### **Microsoft.Windows.UpdateCsp.ExecuteRollBackQualityStarted**

The Execute Rollback Quality Started event sends basic information on the start process to provide information that the Quality Rollback has started.

### **Microsoft.Windows.UpdateCsp.ExecuteRollBackQualitySucceeded**

The Execute Rollback Quality Succeed event sends basic telemetry on the success of the rollback of the Quality/LCU builds. This functionality supports our feature by providing insights to IT Admins of the success of the Quality rollback.

## Windows Update Delivery Optimization events

### **Microsoft.OSG.DU.DeliveryOptClient.DownloadStarted**

This event sends data describing the start of a new download to enable Delivery Optimization. It's used to understand and address problems regarding downloads.

The following fields are available:

- **background** If the download is happening in the background
- **bytesRequested** Number of bytes requested for download.
- **cdnUrl** Url of the source CDN
- **costFlags** Network cost flags
- **deviceProfile** Identifies the usage or form factor (Desktop, Xbox, VM, etc)
- **diceRoll** Random number used for determining if a client will use peering

- **doClientVersion** Version of the Delivery Optimization client
- **doErrorCode** Delivery Optimization error code returned
- **downloadMode** DownloadMode used (CdnOnly = 0, Lan = 1, Group = 2, Internet = 3, Simple = 99, Bypass = 100)
- **downloadModeSrc** Source of the DownloadMode setting (KvsProvider: 0, GeoProvider: 1, GeoVerProvider: 2, CpProvider: 3, DiscoveryProvider: 4, RegistryProvider: 5, GroupPolicyProvider: 6, MdmProvider: 7, SettingsProvider: 8, InvalidProviderType: 9)
- **errorCode** Error code returned
- **experimentId** Used to correlate client/services calls that are part of the same test during A/B testing
- **fileID** ID of the File being downloaded
- **filePath** Path to where the downloaded file will be written
- **fileSize** Total filesize of the file that was downloaded
- **fileSizeCaller** Value for total file size provided by our caller
- **groupId** ID for the group
- **isVpn** If the machine is connected to a Virtual Private Network
- **jobID** Identifier for the Windows Update Job
- **peerID** ID for this Delivery Optimization client
- **predefinedCallerName** Name of the API caller
- **sessionID** ID for the file download session
- **setConfigs** ID of the update being downloaded
- **updateID** ID for the file download session
- **usedMemoryStream** If the download is using memory streaming in App downloads
- **callerName** Name of the API Caller
- **minDiskSizeGB** The minimum disk size policy set for the device to allow Peering with Delivery Optimization
- **minDiskSizePolicyEnforced** If there is an enforced minimum disk size requirement for peering
- **minFileSizePolicy** The minimum file size policy set for the device to allow Peering with Delivery Optimization
- **scenarioID** ID for the Scenario
- **isEncrypted** Whether the download is encrypted

## Windows Update events

### Microsoft.Windows.Update.DeviceUpdateAgent.UpdateAgentAnalysisSummary

This event collects information regarding the state of devices and drivers on the system following a reboot after the install phase of the new device manifest UUP (Unified Update Platform) update scenario which is used to install a device manifest describing a set of driver packages.

The following fields are available:

- **activated** Whether the entire device manifest update is considered activated and in use.
- **analysisErrorCount** How many driver packages that could not be analyzed because errors were hit during the analysis.
- **flightId** Unique ID for each flight.
- **missingDriverCount** How many driver packages that were delivered by the device manifest that are missing from the system.
- **missingUpdateCount** How many updates that were part of the device manifest that are missing from the system.
- **objectId** Unique value for each diagnostics session.
- **publishedCount** How many drivers packages that were delivered by the device manifest that are published and available to be used on devices.

- **relatedCV** Correlation vector value generated from the latest USO scan.
- **scenarioId** Indicates the update scenario.
- **sessionId** Unique value for each update session.
- **summary** A summary string that contains some basic information about driver packages that are part of the device manifest and any devices on the system that those driver packages match on.
- **summaryAppendError** A Boolean indicating if there was an error appending more information to the summary string.
- **truncatedDeviceCount** How many devices are missing from the summary string due to there not being enough room in the string.
- **truncatedDriverCount** How many driver packages are missing from the summary string due to there not being enough room in the string.
- **unpublishedCount** How many drivers packages that were delivered by the device manifest that are still unpublished and unavailable to be used on devices.
- **updateId** Unique ID for each Update.

### **Microsoft.Windows.Update.DeviceUpdateAgent.UpdateAgentCommit**

This event collects information regarding the final commit phase of the new device manifest UUP (Unified Update Platform) update scenario, which is used to install a device manifest describing a set of driver packages

The following fields are available:

- **errorCode** The error code returned for the current session initialization
- **flightId** The unique identifier for each flight
- **objectId** The unique GUID for each diagnostics session
- **relatedCV** A correlation vector value, generated from the latest USO scan
- **result** Outcome of the initialization of the session
- **scenarioId** Identifies the Update scenario
- **sessionId** The unique value for each update session
- **updateId** The unique identifier for each Update

### **Microsoft.Windows.Update.DeviceUpdateAgent.UpdateAgentDownloadRequest**

This event collects information regarding the download request phase of the new device manifest UUP (Unified Update Platform) update scenario, which is used to install a device manifest describing a set of driver packages

The following fields are available:

- **deletedCorruptFiles** Indicates if UpdateAgent found any corrupt payload files and whether the payload was deleted
- **errorCode** The error code returned for the current session initialization
- **flightId** The unique identifier for each flight
- **objectId** Unique value for each Update Agent mode
- **packageCountOptional** Number of optional packages requested
- **packageCountRequired** Number of required packages requested
- **packageCountTotal** Total number of packages needed
- **packageCountTotalCanonical** Total number of canonical packages
- **packageCountTotalDiff** Total number of diff packages
- **packageCountTotalExpress** Total number of express packages
- **packageSizeCanonical** Size of canonical packages in bytes
- **packageSizeDiff** Size of diff packages in bytes
- **packageSizeExpress** Size of express packages in bytes
- **rangeRequestState** Represents the state of the download range request

- **relatedCV** Correlation vector value generated from the latest USO scan
- **result** Result of the download request phase of update
- **scenarioId** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **sessionId** Unique value for each Update Agent mode attempt
- **updateId** Unique ID for each update

#### **Microsoft.Windows.Update.DeviceUpdateAgent.UpdateAgentInitialize**

This event sends data for initializing a new update session for the new device manifest UUP (Unified Update Platform) update scenario, which is used to install a device manifest describing a set of driver packages

The following fields are available:

- **errorCode** The error code returned for the current initialize phase
- **flightId** The unique identifier for each flight
- **flightMetadata** Contains the FlightId and the build being flighted
- **objectId** Unique value for each Update Agent mode
- **relatedCV** Correlation vector value generated from the latest USO scan
- **result** Result of the initialize phase of update. 0 = Succeeded, 1 = Failed, 2 = Cancelled, 3 = Blocked, 4 = BlockCancelled
- **scenarioId** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **sessionData** Contains instructions to update agent for processing FODs and DUICs (Null for other scenarios)
- **sessionId** Unique value for each Update Agent mode attempt
- **updateId** Unique ID for each update

#### **Microsoft.Windows.Update.DeviceUpdateAgent.UpdateAgentInstall**

This event collects information regarding the install phase of the new device manifest UUP (Unified Update Platform) update scenario, which is used to install a device manifest describing a set of driver packages

The following fields are available:

- **errorCode** The error code returned for the current install phase
- **flightId** The unique identifier for each flight
- **objectId** Unique value for each Update Agent mode
- **relatedCV** Correlation vector value generated from the latest scan
- **result** Result of the install phase of update. 0 = Succeeded 1 = Failed, 2 = Cancelled, 3 = Blocked, 4 = BlockCancelled
- **scenarioId** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **sessionId** Unique value for each Update Agent mode attempt
- **updateId** Unique ID for each update

#### **Microsoft.Windows.Update.DeviceUpdateAgent.UpdateAgentModeStart**

This event sends data for the start of each mode during the process of updating device manifest assets via the UUP (Unified Update Platform) update scenario, which is used to install a device manifest describing a set of driver packages.

The following fields are available:

- **flightId** The unique identifier for each flight
- **mode** Indicates that the Update Agent mode that has started. 1 = Initialize, 2 = DownloadRequest, 3 = Install, 4 = Commit

- **objectId** Unique value for each Update Agent mode
- **relatedCV** Correlation vector value generated from the latest scan
- **scenarioId** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **sessionId** Unique value for each Update Agent mode attempt
- **updateId** Unique ID for each update

#### **Microsoft.Windows.Update.NotificationUx.DialogNotificationToBeDisplayed**

Dialog notification about to be displayed to user.

The following fields are available:

- **AcceptAutoModeLimit** Maximum number of days for a device to automatically enter Auto Reboot mode
- **AutoToAutoFailedLimit** Maximum number of days for Auto Reboot mode to fail before RebootFailed dialog will be shown
- **DeviceLocalTime** Time of dialog shown on local device
- **EngagedModeLimit** Number of days to switch between DTE dialogs
- **EnterAutoModeLimit** Maximum number of days for a device to enter Auto Reboot mode
- **ETag** OneSettings versioning value
- **IsForcedEnabled** Is Forced Reboot mode enabled for this device?
- **IsUltimateForcedEnabled** Is Ultimate Forced Reboot mode enabled for this device?
- **NotificationUxState** Which dialog is shown (ENUM)?
- **NotificationUxStateString** Which dialog is shown (string mapping)?
- **RebootUxState** Engaged/Auto/Forced/UltimeForced
- **RebootUxStateString** Engaged/Auto/Forced/UltimeForced
- **RebootVersion** Version of DTE
- **SkipToAutoModeLimit** The minimum length of time to pass in reboot pending before a machine can be put into auto mode
- **UpdateId** The ID of the update that is pending reboot to finish installation
- **UpdateRevision** The revision of the update that is pending reboot to finish installation
- **UtcTime** The Coordinated Universal Time when the dialog notification will be displayed.
- **DaysSinceRebootRequired** Number of days since reboot was required.

#### **Microsoft.Windows.Update.NotificationUx.EnhancedEngagedRebootAcceptAutoDialog**

Enhanced Engaged reboot accept auto dialog was displayed.

The following fields are available:

- **DeviceLocalTime** Local time of the device sending the event
- **ETag** OneSettings ETag
- **ExitCode** Dialog exit code - user response
- **RebootVersion** Reboot flow version
- **UpdateId** Id of pending update
- **UpdateRevision** Revision number of the pending update
- **UserResponseString** User response to the reboot dialog
- **UtcTime** The Coordinated Universal Time that dialog was displayed

#### **Microsoft.Windows.Update.NotificationUx.EnhancedEngagedRebootFirstReminderDialog**

Enhanced Engaged reboot first reminder dialog was displayed.

The following fields are available:

- **DeviceLocalTime** Time of dialog shown on local device
- **ETag** OneSettings versioning value
- **ExitCode** Indicates how users exited the dialog
- **RebootVersion** Version of DTE
- **UpdateId** The id of the update that is pending reboot to finish installation
- **UpdateRevision** The revision of the update that is pending reboot to finish installation
- **UserResponseString** The option that user chose on this dialog
- **UtcTime** The Coordinated Universal Time that dialog was displayed

#### **Microsoft.Windows.Update.NotificationUx.EnhancedEngagedRebootForcedPrecursorDialog**

Enhanced Engaged reboot forced precursor dialog was displayed.

The following fields are available:

- **DeviceLocalTime** Time of dialog shown on local device
- **ETag** OneSettings versioning value
- **ExitCode** Indicates how users exited the dialog
- **RebootVersion** Version of DTE
- **UpdateId** The id of the update that is pending reboot to finish installation
- **UpdateRevision** The revision of the update that is pending reboot to finish installation
- **UserResponseString** The option that user chose on this dialog
- **UtcTime** The Coordinated Universal Time that dialog was displayed

#### **Microsoft.Windows.Update.NotificationUx.EnhancedEngagedRebootForcedWarningDialog**

Enhanced Engaged forced warning dialog was displayed.

The following fields are available:

- **DeviceLocalTime** Time of dialog shown on local device
- **ETag** OneSettings versioning value
- **ExitCode** Indicates how users exited the dialog
- **RebootVersion** Version of DTE
- **UpdateId** The id of the update that is pending reboot to finish installation
- **UpdateRevision** The revision of the update that is pending reboot to finish installation
- **UserResponseString** The option that user chose on this dialog
- **UtcTime** The Coordinated Universal Time that dialog was displayed

#### **Microsoft.Windows.Update.NotificationUx.EnhancedEngagedRebootRebootFailedDialog**

Enhanced Engaged reboot reboot failed dialog was displayed.

The following fields are available:

- **DeviceLocalTime** Dialog exit code - user response
- **ETag** OneSettings versioning value
- **ExitCode** Indicates how users exited the dialog
- **RebootVersion** Version of DTE
- **UpdateId** The ID of the update that is pending reboot to finish installation
- **UpdateRevision** The revision of the update that is pending reboot to finish installation
- **UserResponseString** The option that user chose on this dialog
- **UtcTime** The Coordinated Universal Time that dialog was displayed

#### **Microsoft.Windows.Update.NotificationUx.EnhancedEngagedRebootRebootImminentDialog**

Enhanced Engaged reboot imminent dialog was displayed.

The following fields are available:

- **DeviceLocalTime** Time of dialog shown on local device
- **ETag** OneSettings versioning value
- **ExitCode** Indicates how users exited the dialog
- **RebootVersion** Version of DTE
- **UpdateId** The ID of the update that is pending reboot to finish installation
- **UpdateRevision** The revision of the update that is pending reboot to finish installation
- **UserResponseString** The option that user chose on this dialog
- **UtcTime** The Coordinated Universal Time that dialog was displayed

#### **Microsoft.Windows.Update.NotificationUx.EnhancedEngagedRebootSecondReminderDialog**

Enhanced Engaged reboot second reminder dialog was displayed.

The following fields are available:

- **DeviceLocalTime** Time of dialog shown on local device
- **ETag** OneSettings versioning value
- **ExitCode** Indicates how users exited the dialog
- **RebootVersion** Version of DTE
- **UpdateId** The ID of the update that is pending reboot to finish installation
- **UpdateRevision** The revision of the update that is pending reboot to finish installation
- **UserResponseString** The option that user chose on this dialog
- **UtcTime** The Coordinated Universal Time that dialog was displayed

#### **Microsoft.Windows.Update.NotificationUx.EnhancedEngagedRebootThirdReminderDialog**

Enhanced Engaged reboot third reminder dialog was displayed.

The following fields are available:

- **DeviceLocalTime** Time of dialog shown on local device
- **ETag** OneSettings versioning value
- **ExitCode** Indicates how users exited the dialog
- **RebootVersion** Version of DTE
- **UpdateId** The ID of the update that is pending reboot to finish installation
- **UpdateRevision** The revision of the update that is pending reboot to finish installation
- **UserResponseString** The option that user chose on this dialog
- **UtcTime** The Coordinated Universal Time that dialog was displayed

#### **Microsoft.Windows.Update.NotificationUx.RebootScheduled**

Indicates when a reboot is scheduled by the system or a user for a security, quality, or feature update

The following fields are available:

- **activeHoursApplicable** True, If Active Hours applicable on this device. False, otherwise
- **rebootArgument** Argument for the reboot task. It also represents specific reboot related action
- **rebootOutsideOfActiveHours** True, if a reboot is scheduled outside of active hours. False, otherwise
- **rebootScheduledByUser** True, if a reboot is scheduled by user. False, if a reboot is scheduled automatically
- **rebootState** The state of the reboot
- **revisionNumber** Revision number of the update that is getting installed with this reboot
- **scheduledRebootTime** Time of the scheduled reboot

- **scheduledRebootTimeInUTC** Time of the scheduled reboot in Coordinated Universal Time
- **updateId** ID of the update that is getting installed with this reboot
- **wuDeviceid** Unique device ID used by Windows Update
- **IsEnhancedEngagedReboot** Whether this is an Enhanced Engaged reboot

#### **Microsoft.Windows.Update.Orchestrator.ActivityRestrictedByActiveHoursPolicy**

A policy is present that may restrict update activity to outside of active hours.

The following fields are available:

- **activeHoursEnd** The end of the active hours window
- **activeHoursStart** The start of the active hours window
- **wuDeviceid** Device ID

#### **Microsoft.Windows.Update.Orchestrator.BlockedByActiveHours**

Update activity blocked due to active hours being currently active.

The following fields are available:

- **blockReason** The current state of the update process
- **updatePhase** The current state of the update process
- **wuDeviceid** Device ID
- **activeHoursEnd** The end of the active hours window
- **activeHoursStart** The start of the active hours window

#### **Microsoft.Windows.Update.Orchestrator.BlockedByBatteryLevel**

Update activity blocked due to low battery level.

The following fields are available:

- **batteryLevel** The current battery charge capacity
- **batteryLevelThreshold** The battery capacity threshold to stop update activity
- **blockReason** The current state of the update process
- **updatePhase** The current state of the update process
- **wuDeviceid** Device ID

#### **Microsoft.Windows.Update.Orchestrator.CommitFailed**

This events tracks when a device needs to restart after an update but did not.

The following fields are available:

- **errorCode** The error code that was returned.
- **wuDeviceid** The Windows Update device GUID.

#### **Microsoft.Windows.Update.Orchestrator.DTUCompletedWhenWuFlightPendingCommit**

Event to indicate that DTU completed installation of the ESD, when WU was already Pending Commit of the feature update.

The following fields are available:

- **wuDeviceid** Device ID used by WU

#### **Microsoft.Windows.Update.Orchestrator.DTUEnabled**

Inbox DTU functionality enabled.

The following fields are available:



- **wuDeviceid** Device ID.

#### **Microsoft.Windows.Update.Orchestrator.DTUInitiated**

Inbox DTU functionality initiated.

The following fields are available:

- **dtuErrorCode** Return code from creating the DTU Com Server.
- **isDtuApplicable** Determination of whether DTU is applicable to the machine it is running on.
- **wuDeviceid** Return code from creating the DTU Com Server.

#### **Microsoft.Windows.Update.Orchestrator.DeferRestart**

Indicates that a restart required for installing updates was postponed.

The following fields are available:

- **displayNeededReason** Semicolon-separated list of reasons reported for display needed
- **eventScenario** Indicates the purpose of the event - whether because scan started, succeeded, failed, etc
- **filteredDeferReason** The raised reason that the USO did not restart (e.g. user active, low battery) that were ignorable
- **gameModeReason** Name of the executable that caused the game mode state check to trigger.
- **ignoredReason** Semicolon-separated list of reasons that were intentionally ignored.
- **revisionNumber** Update ID revision number
- **systemNeededReason** Semicolon-separated list of reasons reported for system needed.
- **updateId** Update ID
- **updateScenarioType** Update session type
- **wuDeviceid** Windows Update Device GUID
- **raisedDeferReason** The reason that the USO did not restart (e.g. user active, low battery)

#### **Microsoft.Windows.Update.Orchestrator.Detection**

A scan for an update occurred.

The following fields are available:

- **detectionBlockingPolicy** State of update action
- **detectionBlockreason** Reason for detection not completing.
- **eventScenario** End to end update session ID, or indicates the purpose of sending this event - whether because the software distribution just started installing content, or whether it was cancelled, succeeded, or failed.
- **interactive** Identifies if session is User Initiated.
- **scanTriggerSource** Source of the triggered scan.
- **updateScenarioType** The update session type.
- **wuDeviceid** Unique device ID used by Windows Update.
- **detectionRetryMode** If we retry to scan
- **errorCode** The returned error code.
- **deferReason** Reason for postponing detection
- **flightID** Flight info
- **revisionNumber** Update version
- **updateId** Update ID - GUID
- **networkStatus** Error info

#### **Microsoft.Windows.Update.Orchestrator.DisplayNeeded**

Reboot postponed due to needing a display

The following fields are available:

- **displayNeededReason** Reason the display is needed
- **eventScenario** Indicates the purpose of sending this event - whether because the software distribution just started checking for content, or whether it was cancelled, succeeded, or failed
- **rebootOutsideOfActiveHours** Indicates the timing that the reboot was to occur to ensure the correct update process and experience is provided to keep Windows up to date
- **revisionNumber** Revision number of the update
- **updateId** Update ID
- **updateScenarioType** The update session type
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date
- **wuDeviceid** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue

#### **Microsoft.Windows.Update.Orchestrator.Download**

This event sends launch data for a Windows Update download to help keep Windows up to date.

The following fields are available:

- **deferReason** Reason for download not completing
- **errorCode** An error code represented as a hexadecimal value
- **eventScenario** End to end update session ID.
- **flightID** Unique update ID.
- **interactive** Identifies if session is user initiated.
- **revisionNumber** Update revision number.
- **updateId** Update ID.
- **updateScenarioType** The update session type.
- **wuDeviceid** Unique device ID used by Windows Update.

#### **Microsoft.Windows.Update.Orchestrator.Escalation**

Event sent when USO takes an Escalation action on device.

The following fields are available:

- **configVersion** Escalation config version on device
- **escalationAction** Indicate the specific escalation action that took place on device
- **updateClassificationGUID** GUID of the update the device is offered
- **updateId** ID of the update the device is offered
- **wuDeviceid** Device ID used by WU

#### **Microsoft.Windows.Update.Orchestrator.EscalationRiskLevels**

Event sent during update scan, download, install. Indicates that the device is at risk of being out-of-date.

The following fields are available:

- **configVersion** Escalation config version on device
- **downloadElapsedTime** How long since the download is required on device
- **downloadRiskLevel** At-risk level of download phase
- **installElapsedTime** How long since the install is required on device
- **installRiskLevel** At-risk level of install phase
- **isSediment** WaaSmedic's assessment of whether is device is at risk or not
- **scanElapsedTime** How long since the scan is required on device

- **scanRiskLevel** At-risk level of scan phase
- **wuDeviceid** Device id used by WU

#### **Microsoft.Windows.Update.Orchestrator.EscalationsRefreshFailed**

USO has a set of escalation actions to prevent a device from becoming out-of-date, and the actions are triggered based on the Escalation config that USO obtains from OneSettings. This event is sent when USO fails to refresh the escalation config from OneSettings.

The following fields are available:

- **configVersion** Current escalation config version on device
- **errorCode** Error code for the refresh failure
- **wuDeviceid** Device ID used by WU

#### **Microsoft.Windows.Update.Orchestrator.FlightInapplicable**

The Update is no longer Applicable to this device

The following fields are available:

- **EventPublishedTime** Flight specific info
- **flightID** Update ID revision number
- **revisionNumber** Update ID - GUID
- **updateId** Update session type
- **updateScenarioType** Last status of update
- **UpdateStatus** Is UUP fallback configured?
- **UUPFallbackConfigured** Windows Update Device GUID
- **wuDeviceid** Windows Update Device GUID

#### **Microsoft.Windows.Update.Orchestrator.GameActive**

This event indicates that an enabled GameMode process prevented the device from restarting to complete an update

The following fields are available:

- **eventScenario** Indicates the purpose of sending this event - whether because the software distribution just started checking for content, or whether it was cancelled, succeeded, or failed
- **gameModeReason** Name of the enabled GameMode process that prevented the device from restarting to complete an update
- **wuDeviceid** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue

#### **Microsoft.Windows.Update.Orchestrator.InitiatingReboot**

This event sends data about an Orchestrator requesting a reboot from power management to help keep Windows up to date.

The following fields are available:

- **EventPublishedTime** Time of the event.
- **flightID** Unique update ID
- **interactive** Indicates the reboot initiation stage of the update process was entered as a result of user action or not.
- **rebootOutsideOfActiveHours** Indicates the timing that the reboot was to occur to ensure the correct update process and experience is provided to keep Windows up to date.
- **revisionNumber** Revision number of the update.

- **updateId** Update ID.
- **updateScenarioType** The update session type.
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date.
- **wuDeviceid** Unique device ID used by Windows Update.

### Microsoft.Windows.Update.Orchestrator.Install

This event sends launch data for a Windows Update install to help keep Windows up to date.

The following fields are available:

- **batteryLevel** Current battery capacity in mWh or percentage left.
- **deferReason** Reason for install not completing.
- **eventScenario** End to end update session ID.
- **interactive** Identifies if session is user initiated.
- **rebootOutsideOfActiveHours** Indicates the timing that the reboot was to occur to ensure the correct update process and experience is provided to keep Windows up to date.
- **updateScenarioType** The update session type.
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date.
- **wuDeviceid** Unique device ID used by Windows Update.
- **flightID** Unique update ID
- **flightUpdate** Flight update
- **ForcedRebootReminderSet** A boolean value that indicates if a forced reboot will happen for updates.
- **installRebootinitiatetime** The time it took for a reboot to be attempted.
- **minutesToCommit** The time it took to install updates.
- **revisionNumber** Update revision number.
- **updateId** Update ID.
- **errorCode** The error code represented by a hexadecimal value.
- **installCommitfailedtime** The time it took for a reboot to happen but the upgrade failed to progress.

### Microsoft.Windows.Update.Orchestrator.PostInstall

Event sent after Update install completes.

The following fields are available:

- **batteryLevel** Battery level percentage
- **bundleId** Update ID - GUID
- **bundleRevisionnumber** Update ID revision number
- **errorCode** Error value
- **eventScenario** State of update action
- **sessionType** Update session type
- **wuDeviceid** Windows Update device GUID
- **flightID** The flight ID of the device
- **updateScenarioType** The scenario type of this update

### Microsoft.Windows.Update.Orchestrator.PowerMenuOptionsChanged

This event is sent when the options in power menu changed, usually due to an update pending reboot, or after a update is installed.

The following fields are available:

- **powermenuNewOptions** The new options after the power menu changed
- **powermenuOldOptions** The old options before the power menu changed
- **rebootPendingMinutes** If the power menu changed because a reboot is pending due to a update, how long that reboot has been pending
- **wuDeviceid** If the power menu changed because a reboot is pending due to a update, the device ID recorded by WU

#### **Microsoft.Windows.Update.Orchestrator.PreShutdownStart**

This event is generated right before the shutdown and commit operations

The following fields are available:

- **wuDeviceid** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue

#### **Microsoft.Windows.Update.Orchestrator.Progress**

Event sent when the download of a update reaches a milestone change, such as network cost policy changed, a internal phase has completed, or a transient state has changed.

The following fields are available:

- **errorCode** Error info
- **flightID** Flight info
- **interactive** Is USO session interactive or non-interactive?
- **networkCostPolicy** The current network cost policy on device
- **revisionNumber** Update ID revision number
- **updateId** Update ID - GUID
- **updateScenarioType** Update Session type
- **updateState** Subphase of the download
- **UpdateStatus** Subphase of the update
- **wuDeviceid** Device ID

#### **Microsoft.Windows.Update.Orchestrator.RebootFailed**

This event sends information about whether an update required a reboot and reasons for failure to help keep Windows up to date.

The following fields are available:

- **batteryLevel** Current battery capacity in mWh or percentage left.
- **deferReason** Reason for install not completing.
- **EventPublishedTime** The time that the reboot failure occurred.
- **flightID** Unique update ID.
- **rebootOutsideOfActiveHours** Indicates the timing that the reboot was to occur to ensure the correct update process and experience is provided to keep Windows up to date.
- **RebootResults** Hex code indicating failure reason. Typically, we expect this to be a specific USO generated hex code.
- **revisionNumber** Update revision number.
- **updateId** Update ID.
- **updateScenarioType** The update session type.
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date.
- **wuDeviceid** Unique device ID used by Windows Update.

### **Microsoft.Windows.Update.Orchestrator.RestoreRebootTask**

This event sends data indicating that a reboot task is missing unexpectedly on a device and the task is restored because a reboot is still required, to help keep Windows up to date.

The following fields are available:

- **RebootTaskRestoredTime** Time at which this reboot task was restored.
- **wuDeviceid** Device id on which the reboot is restored

### **Microsoft.Windows.Update.Orchestrator.ScanTriggered**

Indicates that Update Orchestrator has started a scan operation.

The following fields are available:

- **errorCode** Error info
- **eventScenario** Indicates the purpose of sending this event
- **interactive** Whether or not the scan is interactive.
- **isScanPastSla** Has the SLA elapsed for scanning?
- **isScanPastTriggerSla** Has the SLA elapsed for triggering a scan?
- **minutesOverScanSla** How many minutes over the scan SLA is the scan?
- **minutesOverScanTriggerSla** How many minutes over the scan trigger SLA is the scan?
- **scanTriggerSource** What caused the scan?
- **updateScenarioType** The type of scenario we are in.
- **wuDeviceid** WU Device ID of the machine.

### **Microsoft.Windows.Update.Orchestrator.SystemNeeded**

This event sends data about why a device is unable to reboot, to help keep Windows up to date.

The following fields are available:

- **eventScenario** End to end update session ID.
- **rebootOutsideOfActiveHours** Indicates the timing that the reboot was to occur to ensure the correct update process and experience is provided to keep Windows up to date.
- **revisionNumber** Update revision number.
- **systemNeededReason** Reason ID
- **updateId** Update ID.
- **updateScenarioType** The update session type.
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date.
- **wuDeviceid** Unique device ID used by Windows Update.

### **Microsoft.Windows.Update.Orchestrator.TerminatedByActiveHours**

Update activity was stopped due to active hours starting.

The following fields are available:

- **activeHoursEnd** The end of the active hours window
- **activeHoursStart** The start of the active hours window
- **updatePhase** The current state of the update process
- **wuDeviceid** Device ID

### **Microsoft.Windows.Update.Orchestrator.TerminatedByBatteryLevel**

Update activity was stopped due to a low battery level.

The following fields are available:

- **batteryLevel** The current battery charge capacity
- **batteryLevelThreshold** The battery capacity threshold to stop update activity
- **updatePhase** The current state of the update process
- **wuDeviceid** Device ID

#### **Microsoft.Windows.Update.Orchestrator.UpdatePolicyCacheRefresh**

This event sends data on whether Update Management Policies were enabled on a device, to help keep Windows up to date.

The following fields are available:

- **configuredPoliciescount** Policy Count
- **policiesNamevaluesource** Policy Name
- **policyCacherefreshtime** Refresh time
- **updateInstalluxsetting** This shows whether a user has set policies via UX option
- **wuDeviceid** Unique device ID used by Windows Update.

#### **Microsoft.Windows.Update.Orchestrator.UpdateRebootRequired**

This event sends data about whether an update required a reboot to help keep Windows up to date.

The following fields are available:

- **flightID** Unique update ID.
- **interactive** Indicates the reboot initiation stage of the update process was entered as a result of user action or not.
- **revisionNumber** Update revision number.
- **updateId** Update ID.
- **updateScenarioType** The update session type.
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date.
- **wuDeviceid** Unique device ID used by Windows Update.

#### **Microsoft.Windows.Update.Ux.MusNotification.RebootNoLongerNeeded**

This event is sent when a security update has successfully completed.

The following fields are available:

- **UtcTime** The Coordinated Universal Time that the restart was no longer needed.

#### **Microsoft.Windows.Update.Ux.MusNotification.RebootRequestReasonsToIgnore**

This event is sent when the reboot can be deferred based on some reasons, before reboot attempts.

The following fields are available:

- **Reason** The reason sent which will cause the reboot to defer.

#### **Microsoft.Windows.Update.Ux.MusNotification.RebootScheduled**

The RebootScheduled event sends basic information for scheduling a update related reboot to facilitate the flow of getting security updates and keeping Windows up to date.

The following fields are available:

- **activeHoursApplicable** Whether Active Hours applies.
- **rebootArgument** The reboot arguments

- **rebootOutsideOfActiveHours** If reboot was outside of Active Hours
- **rebootScheduledByUser** If the reboot was scheduled by the user, or the system.
- **rebootState** Which state the reboot is in
- **revisionNumber** Revision number of the OS
- **scheduledRebootTime** Time the reboot was scheduled for.
- **scheduledRebootTimeInUTC** Time the reboot was scheduled for in UTC
- **updateId** UpdateId to identify which update is being scheduled.
- **wuDeviceId** Unique DeviceID
- **IsEnhancedEngagedReboot** If Enhanced reboot was enabled.

#### **Microsoft.Windows.Update.Ux.MusNotification.UxBrokerFirstReadyToReboot**

This event is fired the first time when the reboot is required.

#### **Microsoft.Windows.Update.Ux.MusNotification.UxBrokerScheduledTask**

This event is sent when MUSE broker schedules a task.

The following fields are available:

- **TaskArgument** The arguments with which the task is scheduled.
- **TaskName** Name of the task.

## Windows Update mitigation events

#### **Mitigation360Telemetry.MitigationCustom.CleanupSafeOsImages**

This event sends data specific to the CleanupSafeOsImages mitigation used for OS Updates.

The following fields are available:

- **ClientId** In the WU scenario, this will be the WU client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **FlightId** Unique identifier for each flight.
- **InstanceId** Unique GUID that identifies each instances of setuphost.exe.
- **MitigationScenario** The update scenario in which the mitigation was executed.
- **MountedImageCount** Number of mounted images.
- **MountedImageMatches** Number of mounted images that were under %systemdrive%\$Windows.~BT.
- **MountedImagesFailed** Number of mounted images under %systemdrive%\$Windows.~BT that could not be removed.
- **MountedImagesRemoved** Number of mounted images under %systemdrive%\$Windows.~BT that were successfully removed.
- **MountedImagesSkipped** Number of mounted images that were not under %systemdrive%\$Windows.~BT.
- **RelatedCV** Correlation vector value generated from the latest USO scan.
- **Result** HRESULT of this operation.
- **ScenarioId** ID indicating the mitigation scenario.
- **ScenarioSupported** Indicates whether the scenario was supported.
- **SessionId** Unique value for each update attempt.
- **UpdateId** Unique ID for each Update.
- **Wuld** Unique ID for the Windows Update client.

#### **Mitigation360Telemetry.MitigationCustom.FixAppXReparsePoints**

This event sends data specific to the FixAppXReparsePoints mitigation used for OS updates.

The following fields are available:



- **ClientId** Unique identifier for each flight.
- **FlightId** Unique GUID that identifies each instances of setuphost.exe.
- **InstanceId** The update scenario in which the mitigation was executed.
- **MitigationScenario** Correlation vector value generated from the latest USO scan.
- **RelatedCV** Number of reparse points that are corrupted but we failed to fix them.
- **ReparsePointsFailed** Number of reparse points that were corrupted and were fixed by this mitigation.
- **ReparsePointsFixed** Number of reparse points that are not corrupted and no action is required.
- **ReparsePointsSkipped** HRESULT of this operation.
- **Result** ID indicating the mitigation scenario.
- **ScenarioId** Indicates whether the scenario was supported.
- **ScenarioSupported** Unique value for each update attempt.
- **SessionId** Unique ID for each Update.
- **UpdateId** Unique ID for the Windows Update client.
- **Wuld** Unique ID for the Windows Update client.

### **Mitigation360Telemetry.MitigationCustom.FixupEditionId**

This event sends data specific to the FixupEditionId mitigation used for OS updates.

The following fields are available:

- **ClientId** In the WU scenario, this will be the WU client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **EditionIdUpdated** Determine whether EditionId was changed.
- **FlightId** Unique identifier for each flight.
- **InstanceId** Unique GUID that identifies each instances of setuphost.exe.
- **MitigationScenario** The update scenario in which the mitigation was executed.
- **ProductEditionId** Expected EditionId value based on GetProductInfo.
- **ProductType** Value returned by GetProductInfo.
- **RegistryEditionId** EditionId value in the registry.
- **RelatedCV** Correlation vector value generated from the latest USO scan.
- **Result** HRESULT of this operation.
- **ScenarioId** ID indicating the mitigation scenario.
- **ScenarioSupported** Indicates whether the scenario was supported.
- **SessionId** Unique value for each update attempt.
- **UpdateId** Unique ID for each update.
- **Wuld** Unique ID for the Windows Update client.

# Windows 10, version 1709 basic level Windows diagnostic events and fields

7/24/2018 • 177 minutes to read • [Edit Online](#)

## Applies to

- Windows 10, version 1709

The Basic level gathers a limited set of information that is critical for understanding the device and its configuration including: basic device information, quality-related information, app compatibility, and Microsoft Store. When the level is set to Basic, it also includes the Security level information.

The Basic level helps to identify problems that can occur on a particular device hardware or software configuration. For example, it can help determine if crashes are more frequent on devices with a specific amount of memory or that are running a particular driver version. This helps Microsoft fix operating system or app problems.

Use this article to learn about diagnostic events, grouped by event area, and the fields within each event. A brief description is provided for each field. Every event generated includes common data, which collects device data.

You can learn more about Windows functional and diagnostic data through these articles:

- [Windows 10, version 1703 basic diagnostic events and fields](#)
- [Manage connections from Windows operating system components to Microsoft services](#)
- [Configure Windows diagnostic data in your organization](#)

## Common data extensions

### Common Data Extensions.App

The following fields are available:

- **expld** Associates a flight, such as an OS flight, or an experiment, such as a web site UX experiment, with an event.
- **userid** The userID as known by the application.
- **env** The environment from which the event was logged.
- **asld** An integer value that represents the app session. This value starts at 0 on the first app launch and increments after each subsequent app launch per boot session.

### Common Data Extensions.CS

The following fields are available:

- **sig** A common schema signature that identifies new and modified event schemas.

### Common Data Extensions.CUET

The following fields are available:

- **stld** Represents the Scenario Entry Point ID. This is a unique GUID for each event in a diagnostic scenario. This used to be Scenario Trigger ID.
- **ald** Represents the ETW ActivityId. Logged via TraceLogging or directly via ETW.
- **rald** Represents the ETW Related ActivityId. Logged via TraceLogging or directly via ETW.
- **op** Represents the ETW Op Code.
- **cat** Represents a bitmask of the ETW Keywords associated with the event.

- **flags** Represents the bitmap that captures various Windows specific flags.
- **cpId** The composer ID, such as Reference, Desktop, Phone, Holographic, Hub, IoT Composer.
- **tickets** A list of strings that represent entries in the HTTP header of the web request that includes this event.
- **bseq** Upload buffer sequence number in the format <buffer identifier>:<sequence number>
- **mon** Combined monitor and event sequence numbers in the format <monitor sequence>:<event sequence>

#### Common Data Extensions.Device

The following fields are available:

- **ver** Represents the major and minor version of the extension.
- **localId** Represents a locally defined unique ID for the device, not the human readable device name. Most likely equal to the value stored at HKLM\Software\Microsoft\SQMClient\Machinelid
- **deviceClass** Represents the classification of the device, the device “family”. For example, Desktop, Server, or Mobile.

#### Common Data Extensions.Envelope

The following fields are available:

- **ver** Represents the major and minor version of the extension.
- **name** Represents the uniquely qualified name for the event.
- **time** Represents the event date time in Coordinated Universal Time (UTC) when the event was generated on the client. This should be in ISO 8601 format.
- **popSample** Represents the effective sample rate for this event at the time it was generated by a client.
- **epoch** Represents the epoch and seqNum fields, which help track how many events were fired and how many events were uploaded, and enables identification of data lost during upload and de-duplication of events on the ingress server.
- **seqNum** Represents the sequence field used to track absolute order of uploaded events. It is an incrementing identifier for each event added to the upload queue. The Sequence helps track how many events were fired and how many events were uploaded and enables identification of data lost during upload and de-duplication of events on the ingress server.
- **iKey** Represents an ID for applications or other logical groupings of events.
- **flags** Represents a collection of bits that describe how the event should be processed by the Connected User Experiences and Telemetry component pipeline. The lowest-order byte is the event persistence. The next byte is the event latency.
- **os** Represents the operating system name.
- **osVer** Represents the OS version, and its format is OS dependent.
- **appId** Represents a unique identifier of the client application currently loaded in the process producing the event; and is used to group events together and understand usage pattern, errors by application.
- **appVer** Represents the version number of the application. Used to understand errors by Version, Usage by Version across an app.
- **cV** Represents the Correlation Vector: A single field for tracking partial order of related diagnostic data events across component boundaries.

#### Common Data Extensions.OS

The following fields are available:

- **ver** Represents the major and minor version of the extension.
- **expId** Represents the experiment ID. The standard for associating a flight, such as an OS flight (pre-release build), or an experiment, such as a web site UX experiment, with an event is to record the flight / experiment IDs in Part A of the common schema.
- **locale** Represents the locale of the operating system.
- **bootId** An integer value that represents the boot session. This value starts at 0 on first boot after OS install and

increments after every reboot.

### **Common Data Extensions.User**

The following fields are available:

- **ver** Represents the major and minor version of the extension.
- **localId** Represents a unique user identity that is created locally and added by the client. This is not the user's account ID.

### **Common Data Extensions.XBL**

The following fields are available:

- **nbf** Not before time
- **expId** Expiration time
- **sbx** XBOX sandbox identifier
- **dtv** XBOX device type
- **did** XBOX device ID
- **xid** A list of base10-encoded XBOX User IDs.
- **uts** A bit field, with 2 bits being assigned to each user ID listed in xid. This field is omitted if all users are retail accounts.

### **Common Data Extensions.Consent UI Event**

This User Account Control (UAC) diagnostic data point collects information on elevations that originate from low integrity levels. This occurs when a process running at low integrity level (IL) requires higher (administrator) privileges, and therefore requests for elevation via UAC (consent.exe). By better understanding the processes requesting these elevations, Microsoft can in turn improve the detection and handling of potentially malicious behavior in this path.

The following fields are available:

- **eventType** Represents the type of elevation: If it succeeded, was cancelled, or was auto-approved.
- **splitToken** Represents the flag used to distinguish between administrators and standard users.
- **friendlyName** Represents the name of the file requesting elevation from low IL.
- **elevationReason** Represents the distinction between various elevation requests sources (appcompat, installer, COM, MSI and so on).
- **exeName** Represents the name of the file requesting elevation from low IL.
- **signatureState** Represents the state of the signature, if it signed, unsigned, OS signed and so on.
- **publisherName** Represents the name of the publisher of the file requesting elevation from low IL.
- **cmdLine** Represents the full command line arguments being used to elevate.
- **Hash.Length** Represents the length of the hash of the file requesting elevation from low IL.
- **Hash** Represents the hash of the file requesting elevation from low IL.
- **HashAlgId** Represents the algorithm ID of the hash of the file requesting elevation from low IL.
- **telemetryFlags** Represents the details about the elevation prompt for CEIP data.
- **timeStamp** Represents the time stamp on the file requesting elevation.
- **fileVersionMS** Represents the major version of the file requesting elevation.
- **fileVersionLS** Represents the minor version of the file requesting elevation.

## Common data fields

### **Common Data Fields.MS.Device.DeviceInventory.Change**

These fields are added whenever Ms.Device.DeviceInventoryChange is included in the event.

The following fields are available:

- **syncId** A string used to group StartSync, EndSync, Add, and Remove operations that belong together. This field is unique by Sync period and is used to disambiguate in situations where multiple agents perform overlapping inventories for the same object.
- **objectType** Indicates the object type that the event applies to.
- **Action** The change that was invoked on a device inventory object.
- **inventoryId** Device ID used for Compatibility testing

#### **Common Data Fields.TelClientSynthetic.PrivacySettingsAfterCreatorsUpdate.PreUpgradeSettings**

These fields are added whenever PreUpgradeSettings is included in the event.

The following fields are available:

- **HKLM\_SensorPermissionState.SensorPermissionState** The state of the Location service before the feature update completed.
- **HKLM\_SensorPermissionState.HRESULT** The error code returned when trying to query the Location service for the device.
- **HKCU\_SensorPermissionState.SensorPermissionState** The state of the Location service when a user signs on before the feature update completed.
- **HKCU\_SensorPermissionState.HRESULT** The error code returned when trying to query the Location service for the current user.
- **HKLM\_LocationPlatform.Status** The state of the location platform after the feature update has completed.
- **HKLM\_LocationPlatform.HRESULT** The error code returned when trying to query the location platform for the device.
- **HKLM\_LocationSyncEnabled.AcceptedPrivacyPolicy** The speech recognition state for the device before the feature update completed.
- **HKLM\_LocationSyncEnabled.HRESULT** The error code returned when trying to query the Find My Device service for the device.
- **HKCU\_LocationSyncEnabled.AcceptedPrivacyPolicy** The speech recognition state for the current user before the feature update completed.
- **HKCU\_LocationSyncEnabled.HRESULT** The error code returned when trying to query the Find My Device service for the current user.
- **HKLM\_AllowTelemetry.AllowTelemetry** The state of the Connected User Experiences and Telemetry component for the device before the feature update.
- **HKLM\_AllowTelemetry.HRESULT** The error code returned when trying to query the Connected User Experiences and Telemetry component for the device.
- **HKLM\_TIPC.Enabled** The state of TIPC for the device.
- **HKLM\_TIPC.HRESULT** The error code returned when trying to query TIPC for the device.
- **HKCU\_TIPC.Enabled** The state of TIPC for the current user.
- **HKCU\_TIPC.HRESULT** The error code returned when trying to query TIPC for the current user.
- **HKLM\_FlipAhead.FPEnabled** Is Flip Ahead enabled for the device before the feature update was completed?
- **HKLM\_FlipAhead.HRESULT** The error code returned when trying to query Flip Ahead for the device.
- **HKCU\_FlipAhead.FPEnabled** Is Flip Ahead enabled for the current user before the feature update was completed?
- **HKCU\_FlipAhead.HRESULT** The error code returned when trying to query Flip Ahead for the current user.
- **HKLM\_TailoredExperiences.TailoredExperiencesWithDiagnosticDataEnabled** Is Tailored Experiences with Diagnostics Data enabled for the current user after the feature update had completed?
- **HKCU\_TailoredExperiences.HRESULT** The error code returned when trying to query Tailored Experiences with Diagnostics Data for the current user.

- **HKLM\_AdvertisingID.Enabled** Is the advertising ID enabled for the device?
- **HKLM\_AdvertisingID.HRESULT** The error code returned when trying to query the state of the advertising ID for the device.
- **HKCU\_AdvertisingID.Enabled** Is the advertising ID enabled for the current user?
- **HKCU\_AdvertisingID.HRESULT** The error code returned when trying to query the state of the advertising ID for the user.

### Common Data Fields.TelClientSynthetic.PrivacySettingsAfterCreatorsUpdate.PostUpgradeSettings

These fields are added whenever PostUpgradeSettings is included in the event.

The following fields are available:

- **HKLM\_SensorPermissionState.SensorPermissionState** The state of the Location service after the feature update has completed.
- **HKLM\_SensorPermissionState.HRESULT** The error code returned when trying to query the Location service for the device.
- **HKCU\_SensorPermissionState.SensorPermissionState** The state of the Location service when a user signs on after a feature update has completed.
- **HKCU\_SensorPermissionState.HRESULT** The error code returned when trying to query the Location service for the current user.
- **HKLM\_LocationPlatform.Status** The state of the location platform after the feature update has completed.
- **HKLM\_LocationPlatform.HRESULT** The error code returned when trying to query the location platform for the device.
- **HKLM\_LocationSyncEnabled.AcceptedPrivacyPolicy** The speech recognition state for the device after the feature update has completed.
- **HKLM\_LocationSyncEnabled.HRESULT** The error code returned when trying to query the Find My Device service for the device.
- **HKCU\_LocationSyncEnabled.AcceptedPrivacyPolicy** The speech recognition state for the current user after the feature update has completed.
- **HKCU\_LocationSyncEnabled.HRESULT** The error code returned when trying to query the Find My Device service for the current user.
- **HKLM\_AllowTelemetry.AllowTelemetry** The state of the Connected User Experiences and Telemetry component for the device after the feature update.
- **HKLM\_AllowTelemetry.HRESULT** The error code returned when trying to query the Connected User Experiences and Telemetry component for the device.
- **HKLM\_TIPC.Enabled** The state of TIPC for the device.
- **HKLM\_TIPC.HRESULT** The error code returned when trying to query TIPC for the device.
- **HKCU\_TIPC.Enabled** The state of TIPC for the current user.
- **HKCU\_TIPC.HRESULT** The error code returned when trying to query TIPC for the current user.
- **HKLM\_FlipAhead.FPEnabled** Is Flip Ahead enabled for the device after the feature update has completed?
- **HKLM\_FlipAhead.HRESULT** The error code returned when trying to query Flip Ahead for the device.
- **HKCU\_FlipAhead.FPEnabled** Is Flip Ahead enabled for the current user after the feature update has completed?
- **HKCU\_FlipAhead.HRESULT** The error code returned when trying to query Flip Ahead for the current user.
- **HKLM\_TailoredExperiences.TailoredExperiencesWithDiagnosticDataEnabled** Is Tailored Experiences with Diagnostics Data enabled for the current user after the feature update had completed?
- **HKCU\_TailoredExperiences.HRESULT** The error code returned when trying to query Tailored Experiences with Diagnostics Data for the current user.
- **HKLM\_AdvertisingID.Enabled** Is the advertising ID enabled for the device?
- **HKLM\_AdvertisingID.HRESULT** The error code returned when trying to query the state of the advertising ID

for the device.

- **HKCU\_AdvertisingID.Enabled** Is the advertising ID enabled for the current user?
- **HKCU\_AdvertisingID.HRESULT** The error code returned when trying to query the state of the advertising ID for the user.

## Appraiser events

### Microsoft.Windows.Appraiser.General.RunContext

This event indicates what should be expected in the data payload.

The following fields are available:

- **AppraiserBranch** The source branch in which the currently running version of Appraiser was built.
- **AppraiserProcess** The name of the process that launched Appraiser.
- **AppraiserVersion** The version of the Appraiser file generating the events.
- **Context** Indicates what mode Appraiser is running in. Example: Setup or Diagnostic Data.
- **PCFP** An ID for the system calculated by hashing hardware identifiers.
- **Time** The client time of the event.

### Microsoft.Windows.Appraiser.General.TelemetryRunHealth

A summary event indicating the parameters and result of a diagnostic data run. This allows the rest of the data sent over the course of the run to be properly contextualized and understood, which is then used to keep Windows up-to-date.

The following fields are available:

- **AppraiserBranch** The source branch in which the version of Appraiser that is running was built.
- **AppraiserDataVersion** The version of the data files being used by the Appraiser diagnostic data run.
- **AppraiserProcess** The name of the process that launched Appraiser.
- **AppraiserVersion** The file version (major, minor and build) of the Appraiser DLL, concatenated without dots.
- **AuxFinal** Obsolete, always set to false
- **AuxInitial** Obsolete, indicates if Appraiser is writing data files to be read by the Get Windows 10 app.
- **DeadlineDate** A timestamp representing the deadline date, which is the time until which appraiser will wait to do a full scan.
- **EnterpriseRun** Indicates if the diagnostic data run is an enterprise run, which means appraiser was run from the command line with an extra enterprise parameter.
- **FullSync** Indicates if Appraiser is performing a full sync, which means that full set of events representing the state of the machine are sent. Otherwise, only the changes from the previous run are sent.
- **InventoryFullSync** Indicates if inventory is performing a full sync, which means that the full set of events representing the inventory of machine are sent.
- **PCFP** An ID for the system calculated by hashing hardware identifiers.
- **PerfBackoff** Indicates if the run was invoked with logic to stop running when a user is present. Helps to understand why a run may have a longer elapsed time than normal.
- **PerfBackoffInsurance** Indicates if appraiser is running without performance backoff because it has run with perf backoff and failed to complete several times in a row.
- **RunAppraiser** Indicates if Appraiser was set to run at all. If this is false, it is understood that data events will not be received from this device.
- **RunDate** The date that the diagnostic data run was stated, expressed as a filetime.
- **RunGeneralTel** Indicates if the generaltel.dll component was run. Generaltel collects additional diagnostic data on an infrequent schedule and only from machines at diagnostic data levels higher than Basic.
- **RunOnline** Indicates if appraiser was able to connect to Windows Update and therefore is making decisions

using up-to-date driver coverage information.

- **RunResult** The result of the Appraiser diagnostic data run.
- **SendingUtc** Indicates if the Appraiser client is sending events during the current diagnostic data run.
- **StoreHandlesNotNull** Obsolete, always set to false
- **TelemetrySent** Indicates if diagnostic data was successfully sent.
- **ThrottlingUtc** Indicates if the Appraiser client is throttling its output of CUET events to avoid being disabled. This increases runtime but also diagnostic data reliability.
- **Time** The client time of the event.
- **VerboseMode** Indicates if appraiser ran in Verbose mode, which is a test-only mode with extra logging.
- **WhyFullSyncWithoutTablePrefix** Indicates the reason or reasons that a full sync was generated.

### **Microsoft.Windows.Appraiser.General.EnterpriseScenarioWithDiagTrackServiceRunning**

The event that indicates that Appraiser has been triggered to run an enterprise scenario while the DiagTrack service is installed. This event can only be sent if a special flag is used to trigger the enterprise scenario.

The following fields are available:

- **PCFP** An ID for the system calculated by hashing hardware identifiers.
- **Time** The client time of the event.

### **Microsoft.Windows.Appraiser.General.InventoryApplicationFileAdd**

This event represents the basic metadata about a file on the system. The file must be part of an app and either have a block in the compatibility database or are part of an anti-virus program.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **AvDisplayName** If the app is an anti-virus app, this is its display name.
- **AvProductState** Represents state of antivirus program with respect to whether it's turned on and the signatures are up-to-date.
- **BinaryType** A binary type. Example: UNINITIALIZED, ZERO\_BYTE, DATA\_ONLY, DOS\_MODULE, NE16\_MODULE, PE32\_UNKNOWN, PE32\_I386, PE32\_ARM, PE64\_UNKNOWN, PE64\_AMD64, PE64\_ARM64, PE64\_IA64, PE32\_CLR\_32, PE32\_CLR\_IL, PE32\_CLR\_IL\_PREFER32, PE64\_CLR\_64
- **BinFileVersion** An attempt to clean up FileVersion at the client that tries to place the version into 4 octets.
- **BinProductVersion** An attempt to clean up ProductVersion at the client that tries to place the version into 4 octets.
- **BoeProgramId** If there is no entry in Add/Remove Programs, this is the ProgramID that is generated from the file metadata.
- **CompanyName** The company name of the vendor who developed this file.
- **FileId** A hash that uniquely identifies a file.
- **FileVersion** The File version field from the file metadata under Properties -> Details.
- **HasUpgradeExe** Does the anti-virus app have an upgrade.exe file?
- **IsAv** Is the file an anti-virus reporting EXE?
- **LinkDate** The date and time that this file was linked on.
- **LowerCaseLongPath** The full file path to the file that was inventoried on the device.
- **Name** The name of the file that was inventoried.
- **ProductName** The Product name field from the file metadata under Properties -> Details.
- **ProductVersion** The Product version field from the file metadata under Properties -> Details.
- **ProgramId** A hash of the Name, Version, Publisher, and Language of an application used to identify it.
- **Size** The size of the file (in hexadecimal bytes).



### **Microsoft.Windows.Inventory.Core.InventoryApplicationDriverAdd**

This event represents the drivers that an application installs.

The following fields are available:

- **InventoryVersion** The version of the inventory component
- **Programids** The unique program identifier the driver is associated with.

### **Microsoft.Windows.Inventory.Core.InventoryApplicationDriverStartSync**

This event indicates that a new set of InventoryApplicationDriverStartAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory component.

### **Microsoft.Windows.Appraiser.General.DecisionApplicationFileAdd**

This event sends compatibility decision data about a file to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **BlockAlreadyInbox** The uplevel runtime block on the file already existed on the current OS.
- **BlockingApplication** Are there any application issues that interfere with upgrade due to the file in question?
- **DisplayGenericMessage** Will be a generic message be shown for this file?
- **HardBlock** This file is blocked in the SDB.
- **HasUxBlockOverride** Does the file have a block that is overridden by a tag in the SDB?
- **MigApplication** Does the file have a MigXML from the SDB associated with it that applies to the current upgrade mode?
- **MigRemoval** Does the file have a MigXML from the SDB that will cause the app to be removed on upgrade?
- **NeedsDismissAction** Will the file cause an action that can be dismissed?
- **NeedsInstallPostUpgradeData** After upgrade, the file will have a post-upgrade notification to install a replacement for the app.
- **NeedsNotifyPostUpgradeData** Does the file have a notification that should be shown after upgrade?
- **NeedsReinstallPostUpgradeData** After upgrade, this file will have a post-upgrade notification to reinstall the app.
- **NeedsUninstallAction** The file must be uninstalled to complete the upgrade.
- **SdbBlockUpgrade** The file is tagged as blocking upgrade in the SDB,
- **SdbBlockUpgradeCanReinstall** The file is tagged as blocking upgrade in the SDB. It can be reinstalled after upgrade.
- **SdbBlockUpgradeUntilUpdate** The file is tagged as blocking upgrade in the SDB. If the app is updated, the upgrade can proceed.
- **SdbReinstallUpgrade** The file is tagged as needing to be reinstalled after upgrade in the SDB. It does not block upgrade.
- **SdbReinstallUpgradeWarn** The file is tagged as needing to be reinstalled after upgrade with a warning in the SDB. It does not block upgrade.
- **SoftBlock** The file is softblocked in the SDB and has a warning.

### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoBlockAdd**

This event sends blocking data about any compatibility blocking entries hit on the system that are not directly related to specific applications or devices, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.

### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoBlockAdd**

This event sends compatibility decision data about blocking entries on the system that are not keyed by either applications or devices, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **BlockingApplication** Are there any application issues that interfere with upgrade due to matching info blocks?
- **DisplayGenericMessage** Will a generic message be shown for this block?
- **NeedsUninstallAction** Does the user need to take an action in setup due to a matching info block?
- **SdbBlockUpgrade** Is a matching info block blocking upgrade?
- **SdbBlockUpgradeCanReinstall** Is a matching info block blocking upgrade, but has the can reinstall tag?
- **SdbBlockUpgradeUntilUpdate** Is a matching info block blocking upgrade but has the until update tag?

### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPassiveAdd**

This event sends compatibility database information about non-blocking compatibility entries on the system that are not keyed by either applications or devices, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.

### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPassiveAdd**

This event sends compatibility decision data about non-blocking entries on the system that are not keyed by either applications or devices, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **BlockingApplication** Are there any application issues that interfere with upgrade due to matching info blocks?
- **MigApplication** Is there a matching info block with a mig for the current mode of upgrade?

### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPostUpgradeAdd**

This event sends compatibility database information about entries requiring reinstallation after an upgrade on the system that are not keyed by either applications or devices, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.

### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPostUpgradeAdd**

This event sends compatibility decision data about entries that require reinstall after upgrade. It's used to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **NeedsInstallPostUpgradeData** Will the file have a notification after upgrade to install a replacement for the app?
- **NeedsNotifyPostUpgradeData** Should a notification be shown for this file after upgrade?
- **NeedsReinstallPostUpgradeData** Will the file have a notification after upgrade to reinstall the app?
- **SdbReinstallUpgrade** The file is tagged as needing to be reinstalled after upgrade in the compatibility database (but is not blocking upgrade).

### Microsoft.Windows.Appraiser.General.DatasourceDevicePnpAdd

This event sends compatibility data for a PNP device, to help keep Windows up-to-date.

The following fields are available:

- **ActiveNetworkConnection** Is the device an active network device?
- **AppraiserVersion** The version of the appraiser file generating the events.
- **IsBootCritical** Is the device boot critical?
- **WuDriverCoverage** Is there a driver uplevel for this device according to Windows Update?
- **WuDriverUpdateId** The Windows Update ID of the applicable uplevel driver.
- **WuPopulatedFromId** The expected uplevel driver matching ID based on driver coverage from Windows Update.

### Microsoft.Windows.Appraiser.General.DecisionDevicePnpAdd

This event sends compatibility decision data about a PNP device to help keep Windows up-to-date.

The following fields are available:

- **AssociatedDriverWillNotMigrate** Will the driver associated with this plug-and-play device migrate?
- **AppraiserVersion** The version of the appraiser file generating the events.
- **AssociatedDriverIsBlocked** Is the driver associated with this PNP device blocked?
- **BlockAssociatedDriver** Should the driver associated with this PNP device be blocked?
- **BlockingDevice** Is this PNP device blocking upgrade?
- **BlockUpgradelfDriverBlocked** Is the PNP device both boot critical and does not have a driver included with the OS?
- **BlockUpgradelfDriverBlockedAndOnlyActiveNetwork** Is this PNP device the only active network device?
- **DisplayGenericMessage** Will a generic message be shown during Setup for this PNP device?
- **DriverAvailableInbox** Is a driver included with the operating system for this PNP device?
- **DriverAvailableOnline** Is there a driver for this PNP device on Windows Update?
- **DriverAvailableUplevel** Is there a driver on Windows Update or included with the operating system for this PNP device?
- **DriverBlockOverridden** Is there is a driver block on the device that has been overridden?
- **NeedsDismissAction** Will the user would need to dismiss a warning during Setup for this device?
- **NotRegressed** Does the device have a problem code on the source OS that is no better than the one it would have on the target OS?
- **SdbDeviceBlockUpgrade** Is there an SDB block on the PNP device that blocks upgrade?
- **SdbDriverBlockOverridden** Is there an SDB block on the PNP device that blocks upgrade, but that block was overridden?

### Microsoft.Windows.Appraiser.General.DatasourceDriverPackageAdd

This event sends compatibility database data about driver packages to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.

### Microsoft.Windows.Appraiser.General.DecisionDriverPackageAdd

This event sends decision data about driver package compatibility to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **DriverBlockOverridden** Does the driver package have an SDB block that blocks it from migrating, but that block has been overridden?

- **DriverIsDeviceBlocked** Was the driver package was blocked because of a device block?
- **DriverIsDriverBlocked** Is the driver package blocked because of a driver block?
- **DriverShouldNotMigrate** Should the driver package be migrated during upgrade?
- **SdbDriverBlockOverridden** Does the driver package have an SDB block that blocks it from migrating, but that block has been overridden?

#### **Microsoft.Windows.Appraiser.General.InventorySystemBiosAdd**

This event sends basic metadata about the BIOS to determine whether it has a compatibility block.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **BiosDate** The release date of the BIOS in UTC format.
- **BiosName** The name field from Win32\_BIOS.
- **Manufacturer** The manufacturer field from Win32\_ComputerSystem.
- **Model** The model field from Win32\_ComputerSystem.

#### **Microsoft.Windows.Appraiser.General.SystemMemoryAdd**

This event sends data on the amount of memory on the system and whether it meets requirements, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **Blocking** Is the device from upgrade due to memory restrictions?
- **MemoryRequirementViolated** Was a memory requirement violated?
- **pageFile** The current committed memory limit for the system or the current process, whichever is smaller (in bytes).
- **ram** The amount of memory on the device.
- **ramKB** The amount of memory (in KB).
- **virtual** The size of the user-mode portion of the virtual address space of the calling process (in bytes).
- **virtualKB** The amount of virtual memory (in KB).

#### **Microsoft.Windows.Appraiser.General.DecisionSystemBiosAdd**

This event sends compatibility decision data about the BIOS to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **Blocking** Is the device blocked from upgrade due to a BIOS block?
- **HasBiosBlock** Does the device have a BIOS block?

#### **Microsoft.Windows.Appraiser.General.DatasourceSystemBiosAdd**

This event sends compatibility database information about the BIOS to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **SdbEntries** An array of fields indicating the SDB entries that apply to this BIOS.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorCompareExchangeAdd**

This event sends data indicating whether the system supports the CompareExchange128 CPU requirement, to help keep Windows up to date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **Blocking** Is the upgrade blocked due to the processor?
- **CompareExchange128Support** Does the CPU support CompareExchange128?

#### **Microsoft.Windows.Appraiser.General.SystemProcessorLahfSahfAdd**

This event sends data indicating whether the system supports the LahfSahf CPU requirement, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **Blocking** Is the upgrade blocked due to the processor?
- **LahfSahfSupport** Does the CPU support LAHF/SAHF?

#### **Microsoft.Windows.Appraiser.General.SystemProcessorNxAdd**

This event sends data indicating whether the system supports the NX CPU requirement, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **Blocking** Is the upgrade blocked due to the processor?
- **NXDriverResult** The result of the driver used to do a non-deterministic check for NX support.
- **NXProcessorSupport** Does the processor support NX?

#### **Microsoft.Windows.Appraiser.General.SystemProcessorPrefetchWAdd**

This event sends data indicating whether the system supports the PrefetchW CPU requirement, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **Blocking** Is the upgrade blocked due to the processor?
- **PrefetchWSupport** Does the processor support PrefetchW?

#### **Microsoft.Windows.Appraiser.General.SystemProcessorSse2Add**

This event sends data indicating whether the system supports the SSE2 CPU requirement, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **Blocking** Is the upgrade blocked due to the processor?
- **SSE2ProcessorSupport** Does the processor support SSE2?

#### **Microsoft.Windows.Appraiser.General.SystemWimAdd**

This event sends data indicating whether the operating system is running from a compressed WIM file, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **IsWimBoot** Is the current operating system running from a compressed WIM file?
- **RegistryWimBootValue** The raw value from the registry that is used to indicate if the device is running from a WIM.

### **Microsoft.Windows.Appraiser.General.SystemTouchAdd**

This event sends data indicating whether the system supports touch, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **IntegratedTouchDigitizerPresent** Is there an integrated touch digitizer?
- **MaximumTouches** The maximum number of touch points supported by the device hardware.

### **Microsoft.Windows.Appraiser.General.SystemWindowsActivationStatusAdd**

This event sends data indicating whether the current operating system is activated, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **WindowsIsLicensedApiValue** The result from the API that's used to indicate if operating system is activated.
- **WindowsNotActivatedDecision** Is the current operating system activated?

### **Microsoft.Windows.Appraiser.General.InventoryLanguagePackAdd**

This event sends data about the number of language packs installed on the system, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **HasLanguagePack** Does this device have 2 or more language packs?
- **LanguagePackCount** How many language packs are installed?

### **Microsoft.Windows.Appraiser.General.SystemWlanAdd**

This event sends data indicating whether the system has WLAN, and if so, whether it uses an emulated driver that could block an upgrade, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **Blocking** Is the upgrade blocked because of an emulated WLAN driver?
- **HasWlanBlock** Does the emulated WLAN driver have an upgrade block?
- **WlanEmulatedDriver** Does the device have an emulated WLAN driver?
- **WlanExists** Does the device support WLAN at all?
- **WlanModulePresent** Are any WLAN modules present?
- **WlanNativeDriver** Does the device have a non-emulated WLAN driver?

### **Microsoft.Windows.Appraiser.General.InventoryMediaCenterAdd**

This event sends true/false data about decision points used to understand whether Windows Media Center is used on the system, to help keep Windows up to date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **EverLaunched** Has Windows Media Center ever been launched?
- **HasConfiguredTv** Has the user configured a TV tuner through Windows Media Center?
- **HasExtendedUserAccounts** Are any Windows Media Center Extender user accounts configured?
- **HasWatchedFolders** Are any folders configured for Windows Media Center to watch?
- **IsDefaultLauncher** Is Windows Media Center the default app for opening music or video files?

- **IsPaid** Is the user running a Windows Media Center edition that implies they paid for Windows Media Center?
- **IsSupported** Does the running OS support Windows Media Center?

#### **Microsoft.Windows.Appraiser.General.DecisionMediaCenterAdd**

This event sends decision data about the presence of Windows Media Center, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **BlockingApplication** Is there any application issues that interfere with upgrade due to Windows Media Center?
- **MediaCenterActivelyUsed** If Windows Media Center is supported on the edition, has it been run at least once and are the MediaCenterIndicators are true?
- **MediaCenterIndicators** Do any indicators imply that Windows Media Center is in active use?
- **MediaCenterInUse** Is Windows Media Center actively being used?
- **MediaCenterPaidOrActivelyUsed** Is Windows Media Center actively being used or is it running on a supported edition?
- **NeedsDismissAction** Are there any actions that can be dismissed coming from Windows Media Center?

#### **Microsoft.Windows.Appraiser.General.ChecksumTotalPictureCount**

This event lists the types of objects and how many of each exist on the client device. This allows for a quick way to ensure that the records present on the server match what is present on the client.

The following fields are available:

- **DatasourceApplicationFile\_RS2** The total DatasourceApplicationFile objects targeting Windows 10 version 1703 present on this device.
- **DatasourceDevicePnp\_RS2** The total DatasourceDevicePnp objects targeting Windows 10 version 1703 present on this device.
- **DatasourceDriverPackage\_RS2** The total DatasourceDriverPackage objects targeting Windows 10 version 1703 present on this device.
- **DataSourceMatchingInfoBlock\_RS2** The total DataSourceMatchingInfoBlock objects targeting Windows 10 version 1703 present on this device.
- **DataSourceMatchingInfoPassive\_RS2** The total DataSourceMatchingInfoPassive objects targeting Windows 10 version 1703 present on this device.
- **DataSourceMatchingInfoPostUpgrade\_RS2** The total DataSourceMatchingInfoPostUpgrade objects targeting Windows 10 version 1703 present on this device.
- **DatasourceSystemBios\_RS2** The total DatasourceSystemBios objects targeting Windows 10 version 1703 present on this device.
- **DecisionApplicationFile\_RS2** The total DecisionApplicationFile objects targeting Windows 10 version 1703 present on this device.
- **DecisionDevicePnp\_RS2** The total DecisionDevicePnp objects targeting Windows 10 version 1703 present on this device.
- **DecisionDriverPackage\_RS2** The total DecisionDriverPackage objects targeting Windows 10 version 1703 present on this device.
- **DecisionMatchingInfoBlock\_RS2** The total DecisionMatchingInfoBlock objects targeting Windows 10 version 1703 present on this device.
- **DecisionMatchingInfoPassive\_RS2** The total DecisionMatchingInfoPassive objects targeting Windows 10 version 1703 present on this device.
- **DecisionMatchingInfoPostUpgrade\_RS2** The total DecisionMatchingInfoPostUpgrade objects targeting Windows 10 version 1703 present on this device.
- **DecisionMediaCenter\_RS2** The total DecisionMediaCenter objects targeting Windows 10 version 1703 present on this device.

present on this device.

- **DecisionSystemBios\_RS2** The total DecisionSystemBios objects targeting Windows 10 version 1703 present on this device.
- **InventoryApplicationFile** The total InventoryApplicationFile objects that are present on this device.
- **InventoryLanguagePack** The total InventoryLanguagePack objects that are present on this device.
- **InventoryMediaCenter** The total InventoryMediaCenter objects that are present on this device.
- **InventorySystemBios** The total InventorySystemBios objects that are present on this device.
- **InventoryUplevelDriverPackage** The total InventoryUplevelDriverPackage objects that are present on this device.
- **PCFP** An ID for the system that is calculated by hashing hardware identifiers.
- **SystemMemory** The total SystemMemory objects that are present on this device.
- **SystemProcessorCompareExchange** The total SystemProcessorCompareExchange objects that are present on this device.
- **SystemProcessorLahfSahf** The total SystemProcessorLahfSahf objects that are present on this device.
- **SystemProcessorNx** The total SystemProcessorNx objects that are present on this device.
- **SystemProcessorPrefetchW** The total SystemProcessorPrefetchW objects that are present on this device.
- **SystemProcessorSse2** The total SystemProcessorSse2 objects that are present on this device.
- **SystemTouch** The total SystemTouch objects that are present on this device.
- **SystemWim** The total SystemWim objects that are present on this device
- **SystemWindowsActivationStatus** The total SystemWindowsActivationStatus objects that are present on this device.
- **SystemWlan** The total SystemWlan objects that are present on this device.
- **Wmdrm\_RS2** The total Wmdrm objects targeting Windows 10 version 1703 present on this device.
- **DatasourceApplicationFile\_RS3** "The total DecisionApplicationFile objects targeting the next release of Windows on this device. "
- **DatasourceDevicePnp\_RS3** The total DatasourceDevicePnp objects targeting the next release of Windows on this device.
- **DatasourceDriverPackage\_RS3** The total DatasourceDriverPackage objects targeting the next release of Windows on this device.
- **DataSourceMatchingInfoBlock\_RS3** The total DataSourceMatchingInfoBlock objects targeting the next release of Windows on this device.
- **DataSourceMatchingInfoPassive\_RS3** The total DataSourceMatchingInfoPassive objects targeting the next release of Windows on this device.
- **DataSourceMatchingInfoPostUpgrade\_RS3** The total DataSourceMatchingInfoPostUpgrade objects targeting the next release of Windows on this device.
- **DatasourceSystemBios\_RS3** The total DatasourceSystemBios objects targeting the next release of Windows on this device.
- **DecisionApplicationFile\_RS3** The total DecisionApplicationFile objects targeting the next release of Windows on this device.
- **DecisionDevicePnp\_RS3** The total DecisionDevicePnp objects targeting the next release of Windows on this device.
- **DecisionDriverPackage\_RS3** The total DecisionDriverPackage objects targeting the next release of Windows on this device.
- **DecisionMatchingInfoBlock\_RS3** The total DecisionMatchingInfoBlock objects targeting the next release of Windows on this device.
- **DecisionMatchingInfoPassive\_RS3** The total DataSourceMatchingInfoPassive objects targeting the next release of Windows on this device.
- **DecisionMatchingInfoPostUpgrade\_RS3** The total DecisionMatchingInfoPostUpgrade objects targeting the



next release of Windows on this device.

- **DecisionMediaCenter\_RS3** The total DecisionMediaCenter objects targeting the next release of Windows on this device.
- **DecisionSystemBios\_RS3** The total DecisionSystemBios objects targeting the next release of Windows on this device.
- **Wmdrm\_RS3** The total Wmdrm objects targeting the next release of Windows on this device.

#### **Microsoft.Windows.Appraiser.General.InventoryUplevelDriverPackageStartSync**

This event indicates that a new set of InventoryUplevelDriverPackageAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorLahfSahfStartSync**

This event indicates that a new set of SystemProcessorLahfSahfAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorSse2StartSync**

This event indicates that a new set of SystemProcessorSse2Add events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.InventorySystemBiosStartSync**

This event indicates that a new set of InventorySystemBiosAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionSystemBiosStartSync**

This event indicates that a new set of DecisionSystemBiosAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemMemoryStartSync**

This event indicates that a new set of SystemMemoryAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorCompareExchangeStartSync**

This event indicates that a new set of SystemProcessorCompareExchangeAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorNxStartSync**

This event indicates that a new set of SystemProcessorNxAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorPrefetchWStartSync**

This event indicates that a new set of SystemProcessorPrefetchWAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemWimStartSync**

This event indicates that a new set of SystemWimAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DatasourceSystemBiosStartSync**

This event indicates that a new set of DatasourceSystemBiosAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemTouchStartSync**

This event indicates that a new set of SystemTouchAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DatasourceDriverPackageEndSync**

This event indicates that a full set of DatasourceDriverPackageAdd events has been sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemWlanStartSync**

This event indicates that a new set of SystemWlanAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemWindowsActivationStatusStartSync**

This event indicates that a new set of SystemWindowsActivationStatusAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionMediaCenterStartSync**

This event indicates that a new set of DecisionMediaCenterAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.InventoryMediaCenterStartSync**

This event indicates that a new set of InventoryMediaCenterAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPassiveStartSync**

This event indicates that a new set of DecisionMatchingInfoPassiveAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPassiveStartSync**

This event indicates that a new set of DataSourceMatchingInfoPassiveAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.InventoryApplicationFileStartSync**

This event indicates that a new set of InventoryApplicationFileAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPostUpgradeStartSync**

This event indicates that a new set of DecisionMatchingInfoPostUpgradeAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.WmdrmStartSync**

This event indicates that a new set of WmdrmAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPassiveEndSync**

This event indicates that a full set of DataSourceMatchingInfoPassiveAdd events have been sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoBlockStartSync**

This event indicates that a new set of DecisionMatchingInfoBlockAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DatasourceApplicationFileStartSync**

This event indicates that a new set of DatasourceApplicationFileAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DatasourceDevicePnpStartSync**

This event indicates that a new set of DatasourceDevicePnpAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoBlockStartSync**

This event indicates that a full set of DataSourceMatchingInfoBlockStartAdd events have been sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionApplicationFileStartSync**

This event indicates that a new set of DecisionApplicationFileAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.InventoryLanguagePackStartSync**

This event indicates that a new set of InventoryLanguagePackAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPostUpgradeStartSync**

This event indicates that a new set of DataSourceMatchingInfoPostUpgradeAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionDevicePnpStartSync**

This event indicates that the DecisionDevicePnp object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DatasourceDriverPackageStartSync**

This event indicates that a new set of DatasourceDriverPackageAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionDriverPackageStartSync**

This event indicates that a new set of DecisionDriverPackageAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.WmdrmAdd**

This event sends data about the usage of older digital rights management on the system, to help keep Windows up to date. This data does not indicate the details of the media using the digital rights management, only whether any such files exist. Collecting this data was critical to ensuring the correct mitigation for customers, and should be able to be removed once all mitigations are in place.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **BlockingApplication** Same as NeedsDismissAction
- **NeedsDismissAction** Indicates if a dismissible message is needed to warn the user about a potential loss of data due to DRM deprecation.
- **WmdrmApiResult** Raw value of the API used to gather DRM state.
- **WmdrmCdRipped** Indicates if the system has any files encrypted with personal DRM, which was used for ripped CDs.
- **WmdrmIndicators** WmdrmCdRipped OR WmdrmPurchased
- **WmdrmInUse** WmdrmIndicators AND dismissible block in setup was not dismissed.
- **WmdrmNonPermanent** Indicates if the system has any files with non-permanent licenses.
- **WmdrmPurchased** Indicates if the system has any files with permanent licenses.

#### **Microsoft.Windows.Appraiser.General.InventoryUplevelDriverPackageAdd**

This event is only runs during setup. It provides a listing of the uplevel driver packages that were downloaded before the upgrade. Is critical to understanding if failures in setup can be traced to not having sufficient uplevel drivers before the upgrade.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **BootCritical** Is the driver package marked as boot critical?
- **Build** The build value from the driver package.
- **CatalogFile** The name of the catalog file within the driver package.
- **Class** The device class from the driver package.
- **ClassGuid** The device class GUID from the driver package.
- **Date** The date from the driver package.
- **Inbox** Is the driver package of a driver that is included with Windows?
- **OriginalName** The original name of the INF file before it was renamed. Generally a path under \$WINDOWS.BT\Drivers\DU
- **Provider** The provider of the driver package.
- **PublishedName** The name of the INF file, post-rename.
- **Revision** The revision of the driver package.
- **SignatureStatus** Indicates if the driver package is signed. Unknown:0, Unsigned:1, Signed: 2
- **VersionMajor** The major version of the driver package.
- **VersionMinor** The minor version of the driver package.

#### **Microsoft.Windows.Appraiser.General.GatedRegChange**

This event sends data about the results of running a set of quick-blocking instructions, to help keep Windows up to date.

The following fields are available:

- **NewData** The data in the registry value after the scan completed.
- **OldData** The previous data in the registry value before the scan ran.
- **PCFP** An ID for the system calculated by hashing hardware identifiers.
- **RegKey** The registry key name for which a result is being sent.
- **RegValue** The registry value for which a result is being sent.
- **Time** The client time of the event.

#### **Microsoft.Windows.Appraiser.General.DatasourceApplicationFileRemove**

This event indicates that the DatasourceApplicationFile object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DatasourceDevicePnpRemove**

This event indicates that the DatasourceDevicePnp object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DatasourceDriverPackageRemove**

This event indicates that the DatasourceDriverPackage object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorSse2Remove**

This event indicates that the SystemProcessorSse2 object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.InventoryUplevelDriverPackageRemove**

This event indicates that the InventoryUplevelDriverPackage object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionMediaCenterRemove**

This event indicates that the DecisionMediaCenter object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.InventoryMediaCenterRemove**

This event indicates that the InventoryMediaCenter object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DatasourceSystemBiosRemove**

This event indicates that the DatasourceSystemBios object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionApplicationFileRemove**

This event indicates that the DecisionApplicationFile object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPostUpgradeRemove**

This event indicates that the DecisionMatchingInfoPostUpgrade object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemTouchRemove**

"This event indicates that the SystemTouch object is no longer present. "

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemWindowsActivationStatusRemove**

This event indicates that the SystemWindowsActivationStatus object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemWlanRemove**

"This event indicates that the SystemWlan object is no longer present. "

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPostUpgradeRemove**

This event indicates that the DataSourceMatchingInfoPostUpgrade object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorNxRemove**

This event indicates that the SystemProcessorNx object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoBlockRemove**

This event indicates that the DataSourceMatchingInfoBlock object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionDevicePnpRemove**

This event indicates that the DecisionDevicePnp object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPassiveRemove**

This event Indicates that the DecisionMatchingInfoPassive object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemMemoryRemove**

This event that the SystemMemory object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoBlockRemove**

This event indicates that the DecisionMatchingInfoBlock object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPassiveRemove**

This event indicates that the DataSourceMatchingInfoPassive object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.InventoryApplicationFileRemove**

This event indicates that the InventoryApplicationFile object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemWimRemove**

"This event indicates that the SystemWim object is no longer present. "

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.InventorySystemBiosRemove**

"This event indicates that the InventorySystemBios object is no longer present. "

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.WmdrmRemove**

This event indicates that the Wmdrm object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorLahfSahfRemove**

"This event indicates that the SystemProcessorLahfSahf object is no longer present. "

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.InventoryLanguagePackRemove**

This event indicates that the InventoryLanguagePack object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.



### **Microsoft.Windows.Appraiser.General.DecisionDriverPackageRemove**

This event indicates that the DecisionDriverPackage object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.DecisionSystemBiosRemove**

This event indicates that the DecisionSystemBios object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.SystemProcessorCompareExchangeRemove**

"This event indicates that the SystemProcessorCompareExchange object is no longer present. "

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.SystemProcessorPrefetchWRemove**

This event indicates that the SystemProcessorPrefetchW object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.InventoryDriverBinaryEndSync**

This event indicates that a full set of InventoryDriverBinaryAdd events has been sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

## Census events

### **Census.Battery**

This event sends type and capacity data about the battery on the device, as well as the number of connected standby devices in use, type to help keep Windows up to date.

The following fields are available:

- **InternalBatteryCapabilities** Represents information about what the battery is capable of doing.
- **InternalBatteryCapacityCurrent** Represents the battery's current fully charged capacity in mWh (or relative). Compare this value to `DesignedCapacity` to estimate the battery's wear.
- **InternalBatteryCapacityDesign** Represents the theoretical capacity of the battery when new, in mWh.
- **InternalBatteryNumberOfCharges** Provides the number of battery charges. This is used when creating new products and validating that existing products meets targeted functionality performance.
- **IsAlwaysOnAlwaysConnectedCapable** Represents whether the battery enables the device to be `AlwaysOnAlwaysConnected`. Boolean value.

### **Census.Enterprise**

This event sends data about Azure presence, type, and cloud domain use in order to provide an understanding of the use and integration of devices in an enterprise, cloud, and server environment.

The following fields are available:

- **AzureOSIDPresent** Represents the field used to identify an Azure machine.

- **AzureVMType** Represents whether the instance is Azure VM PAAS, Azure VM IAAS or any other VMs.
- **CDJType** Represents the type of cloud domain joined for the machine.
- **CommercialId** Represents the GUID for the commercial entity which the device is a member of. Will be used to reflect insights back to customers.
- **ContainerType** The type of container, such as process or virtual machine hosted.
- **EnrollmentType** Represents the type of enrollment, such as MDM or Intune, for a particular device.
- **HashedDomain** The hashed representation of the user domain used for login.
- **IsCloudDomainJoined** Is this device joined to an Azure Active Directory (AAD) tenant? true/false
- **IsDERequirementMet** Represents if the device can do device encryption.
- **IsDeviceProtected** Represents if Device protected by BitLocker/Device Encryption
- **IsDomainJoined** Indicates whether a machine is joined to a domain.
- **IsEDPEnabled** Represents if Enterprise data protected on the device.
- **IsMDMEnrolled** Whether the device has been MDM Enrolled or not.
- **MPNId** Returns the Partner ID/MPN ID from Regkey.  
HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\DeployID
- **SCCMClientId** This ID correlate systems that send data to Compat Analytics (OMS) and other OMS based systems with systems in an Enterprise SCCM environment.
- **ServerFeatures** Represents the features installed on a Windows Server. This can be used by developers and administrators who need to automate the process of determining the features installed on a set of server computers.
- **SystemCenterID** The SCCM ID is an anonymized one-way hash of the Active Directory Organization identifier

### Census.App

This event sends version data about the Apps running on this device, to help keep Windows up to date.

The following fields are available:

- **CensusVersion** The version of Census that generated the current data for this device.
- **IEVersion** Retrieves which version of Internet Explorer is running on this device.

### Census.Camera

This event sends data about the resolution of cameras on the device, to help keep Windows up to date.

The following fields are available:

- **FrontFacingCameraResolution** Represents the resolution of the front facing camera in megapixels. If a front facing camera does not exist, then the value is 0.
- **RearFacingCameraResolution** Represents the resolution of the rear facing camera in megapixels. If a rear facing camera does not exist, then the value is 0.

### Census.UserDisplay

This event sends data about the logical/physical display size, resolution and number of internal/external displays, and VRAM on the system, to help keep Windows up to date.

The following fields are available:

- **InternalPrimaryDisplayLogicalDPIX** Retrieves the logical DPI in the x-direction of the internal display.
- **InternalPrimaryDisplayLogicalDPIY** Retrieves the logical DPI in the y-direction of the internal display.
- **InternalPrimaryDisplayPhysicalDPIX** Retrieves the physical DPI in the x-direction of the internal display.
- **InternalPrimaryDisplayPhysicalDPIY** Retrieves the physical DPI in the y-direction of the internal display.
- **InternalPrimaryDisplayResolutionHorizontal** Retrieves the number of pixels in the horizontal direction of the internal display.

- **InternalPrimaryDisplayResolutionVertical** Retrieves the number of pixels in the vertical direction of the internal display.
- **InternalPrimaryDisplaySizePhysicalH** Retrieves the physical horizontal length of the display in mm. Used for calculating the diagonal length in inches .
- **InternalPrimaryDisplaySizePhysicalY** Retrieves the physical vertical length of the display in mm. Used for calculating the diagonal length in inches
- **InternalPrimaryDisplayType** Represents the type of technology used in the monitor, such as Plasma, LED, LCOS, etc.
- **NumberofExternalDisplays** Retrieves the number of external displays connected to the machine
- **NumberofInternalDisplays** Retrieves the number of internal displays in a machine.
- **VRAMDedicated** Retrieves the video RAM in MB.
- **VRAMDedicatedSystem** Retrieves the amount of memory on the dedicated video card.
- **VRAMSharedSystem** Retrieves the amount of RAM memory that the video card can use.

#### Census.Firmware

This event sends data about the BIOS and startup embedded in the device, to help keep Windows up to date.

The following fields are available:

- **FirmwareManufacturer** Represents the manufacturer of the device's firmware (BIOS).
- **FirmwareReleaseDate** Represents the date the current firmware was released.
- **FirmwareType** Represents the firmware type. The various types can be unknown, BIOS, UEFI.
- **FirmwareVersion** Represents the version of the current firmware.

#### Census.Flighting

This event sends Windows Insider data from customers participating in improvement testing and feedback programs, to help keep Windows up-to-date.

The following fields are available:

- **DeviceSampleRate** The diagnostic data sample rate assigned to the device.
- **EnablePreviewBuilds** Used to enable Windows Insider builds on a device.
- **FlightIds** A list of the different Windows Insider builds on this device.
- **FlightingBranchName** The name of the Windows Insider branch currently used by the device.
- **IsFlightsDisabled** Represents if the device is participating in the Windows Insider program.
- **MSA\_Accounts** Represents a list of hashed IDs of the Microsoft Accounts that are flying (pre-release builds) on this device.
- **SSRK** Retrieves the mobile targeting settings.

#### Census.Hardware

This event sends data about the device, including hardware type, OEM brand, model line, model, diagnostic data level setting, and TPM support, to help keep Windows up-to-date.

The following fields are available:

- **ActiveMicCount** The number of active microphones attached to the device.
- **ChassisType** Represents the type of device chassis, such as desktop or low profile desktop. The possible values can range between 1 - 36.
- **ComputerHardwareID** Identifies a device class that is represented by a hash of different SMBIOS fields.
- **D3DMaxFeatureLevel** The supported Direct3D version.
- **DeviceColor** Indicates a color of the device.
- **DeviceForm** Indicates the form as per the device classification.
- **DeviceName** The device name that is set by the user.

- **DigitizerSupport** Is a digitizer supported?
- **DUID** The device unique ID.
- **Gyroscope** Indicates whether the device has a gyroscope.
- **InventoryId** The device ID used for compatibility testing.
- **Magnetometer** Indicates whether the device has a magnetometer.
- **NFCProximity** Indicates whether the device supports NFC.
- **OEMDigitalMarkerFileName** The name of the file placed in the \Windows\system32\drivers directory that specifies the OEM and model name of the device.
- **OEMManufacturerName** The device manufacturer name. The OEMName for an inactive device is not reprocessed even if the clean OEM name is changed at a later date.
- **OEMModelBaseBoard** The baseboard model used by the OEM.
- **OEMModelBaseBoardVersion** Differentiates between developer and retail devices.
- **OEMModelName** The device model name.
- **OEMModelNumber** The device model number.
- **OEMModelSKU** The device edition that is defined by the manufacturer.
- **OEMModelSystemFamily** The system family set on the device by an OEM.
- **OEMModelSystemVersion** The system model version set on the device by the OEM.
- **OEMOptionalIdentifier** A Microsoft assigned value that represents a specific OEM subsidiary.
- **OEMSerialNumber** The serial number of the device that is set by the manufacturer.
- **PhoneManufacturer** The friendly name of the phone manufacturer.
- **PowerPlatformRole** The OEM preferred power management profile. It's used to help to identify the basic form factor of the device.
- **SoCName** The firmware manufacturer of the device.
- **StudyID** Used to identify retail and non-retail device.
- **TelemetryLevel** The diagnostic data level the user has opted into, such as Basic or Enhanced.
- **TelemetryLevelLimitEnhanced** The diagnostic data level for Windows Analytics-based solutions.
- **TelemetrySettingAuthority** Determines who set the diagnostic data level, such as GP, MDM, or the user.
- **TPMVersion** The supported Trusted Platform Module (TPM) on the device. If no TPM is present, the value is 0.
- **VoiceSupported** Does the device have a cellular radio capable of making voice calls?

### Census.Memory

This event sends data about the memory on the device, including ROM and RAM, to help keep Windows up to date.

The following fields are available:

- **TotalPhysicalRAM** Represents the physical memory (in MB).
- **TotalVisibleMemory** Represents the memory that is not reserved by the system.

### Census.Network

This event sends data about the mobile and cellular network used by the device (mobile service provider, network, device ID, and service cost factors), to help keep Windows up to date.

The following fields are available:

- **IMEI0** Represents the International Mobile Station Equipment Identity. This number is usually unique and used by the mobile operator to distinguish different phone hardware. Microsoft does not have access to mobile operator billing data so collecting this data does not expose or identify the user. The two fields represent phone with dual sim coverage.
- **IMEI1** Represents the International Mobile Station Equipment Identity. This number is usually unique and used by the mobile operator to distinguish different phone hardware. Microsoft does not have access to mobile

operator billing data so collecting this data does not expose or identify the user. The two fields represent phone with dual sim coverage.

- **MCC0** Represents the Mobile Country Code (MCC). It used with the Mobile Network Code (MNC) to uniquely identify a mobile network operator. The two fields represent phone with dual sim coverage.
- **MCC1** Represents the Mobile Country Code (MCC). It used with the Mobile Network Code (MNC) to uniquely identify a mobile network operator. The two fields represent phone with dual sim coverage.
- **MEID** Represents the Mobile Equipment Identity (MEID). MEID is a worldwide unique phone ID assigned to CDMA phones. MEID replaces electronic serial number (ESN), and is equivalent to IMEI for GSM and WCDMA phones. Microsoft does not have access to mobile operator billing data so collecting this data does not expose or identify the user.
- **MNC0** Retrieves the Mobile Network Code (MNC). It used with the Mobile Country Code (MCC) to uniquely identify a mobile network operator. The two fields represent phone with dual sim coverage.
- **MNC1** Retrieves the Mobile Network Code (MNC). It used with the Mobile Country Code (MCC) to uniquely identify a mobile network operator. The two fields represent phone with dual sim coverage.
- **MobileOperatorBilling** Represents the telephone company that provides services for mobile phone users.
- **MobileOperatorCommercialized** Represents which reseller and geography the phone is commercialized for. This is the set of values on the phone for who and where it was intended to be used. For example, the commercialized mobile operator code AT&T in the US would be ATT-US.
- **MobileOperatorNetwork0** Represents the operator of the current mobile network that the device is used on. (AT&T, T-Mobile, Vodafone). The two fields represent phone with dual sim coverage.
- **MobileOperatorNetwork1** Represents the operator of the current mobile network that the device is used on. (AT&T, T-Mobile, Vodafone). The two fields represent phone with dual sim coverage.
- **NetworkAdapterGUID** The GUID of the primary network adapter.
- **NetworkCost** Represents the network cost associated with a connection.
- **SPN0** Retrieves the Service Provider Name (SPN). For example, these might be AT&T, Sprint, T-Mobile, or Verizon. The two fields represent phone with dual sim coverage.
- **SPN1** Retrieves the Service Provider Name (SPN). For example, these might be AT&T, Sprint, T-Mobile, or Verizon. The two fields represent phone with dual sim coverage.

## Census.OS

This event sends data about the operating system such as the version, locale, update service configuration, when and how it was originally installed, and whether it is a virtual device, to help keep Windows up to date.

The following fields are available:

- **ActivationChannel** Retrieves the retail license key or Volume license key for a machine.
- **AssignedAccessStatus** The kiosk configuration mode.
- **CompactOS** Indicates if the Compact OS feature from Win10 is enabled.
- **DeveloperUnlockStatus** "Represents if a device has been developer unlocked by the user or Group Policy."
- **DeviceTimeZone** The time zone that is set on the device. Example: Pacific Standard Time
- **GenuineState** Retrieves the ID Value specifying the OS Genuine check.
- **InstallationType** Retrieves the type of OS installation. (Clean, Upgrade, Reset, Refresh, Update).
- **InstallLanguage** The first language installed on the user machine.
- **IsDeviceRetailDemo** Retrieves if the device is running in demo mode.
- **IsEduData** Returns Boolean if the education data policy is enabled.
- **IsPortableOperatingSystem** Retrieves whether OS is running Windows-To-Go
- **IsSecureBootEnabled** Retrieves whether Boot chain is signed under UEFI.
- **LanguagePacks** The list of language packages installed on the device.
- **LicenseStateReason** Retrieves why (or how) a system is licensed or unlicensed. The HRESULT may indicate an error code that indicates a key blocked error, or it may indicate that we are running an OS License granted by

the Microsoft Store.

- **OA3xOriginalProductKey** Retrieves the License key stamped by the OEM to the machine.
- **OSEdition** Retrieves the version of the current OS.
- **OSInstallDateTime** Retrieves the date the OS was installed using ISO 8601 (Date part) == yyyy-mm-dd
- **OSInstallType** Retrieves a numeric description of what install was used on the device i.e. clean, upgrade, refresh, reset, etc
- **OSOOBEDateTime** Retrieves Out of Box Experience (OOBE) Date in Coordinated Universal Time (UTC).
- **OSSKU** Retrieves the Friendly Name of OS Edition.
- **OSSubscriptionStatus** Represents the existing status for enterprise subscription feature for PRO machines.
- **OSSubscriptionTypeId** Returns boolean for enterprise subscription feature for selected PRO machines.
- **OSTimeZoneBiasInMins** Retrieves the time zone set on machine.
- **OSUILocale** Retrieves the locale of the UI that is currently used by the OS.
- **ProductActivationResult** Returns Boolean if the OS Activation was successful.
- **ProductActivationTime** Returns the OS Activation time for tracking piracy issues.
- **ProductKeyID2** Retrieves the License key if the machine is updated with a new license key.
- **RACw7Id** Retrieves the Microsoft Reliability Analysis Component (RAC) Win7 Identifier. RAC is used to monitor and analyze system usage and reliability.
- **ServiceMachineIP** Retrieves the IP address of the KMS host used for anti-piracy.
- **ServiceMachinePort** Retrieves the port of the KMS host used for anti-piracy.
- **ServiceProductKeyID** Retrieves the License key of the KMS
- **SharedPCMode** Returns Boolean for education devices used as shared cart
- **Signature** Retrieves if it is a signature machine sold by Microsoft store.
- **SLICStatus** Whether a SLIC table exists on the device.
- **SLICVersion** Returns OS type/version from SLIC table.

### Census.Processor

This event sends data about the processor (architecture, speed, number of cores, manufacturer, and model number), to help keep Windows up to date.

The following fields are available:

- **KvaShadow** Microcode info of the processor.
- **MMSettingOverride** Microcode setting of the processor.
- **MMSettingOverrideMask** Microcode setting override of the processor.
- **ProcessorArchitecture** Retrieves the processor architecture of the installed operating system.
- **ProcessorClockSpeed** Retrieves the clock speed of the processor in MHz.
- **ProcessorCores** Retrieves the number of cores in the processor.
- **ProcessorIdentifier** The processor identifier of a manufacturer.
- **ProcessorManufacturer** Retrieves the name of the processor's manufacturer.
- **ProcessorModel** Retrieves the name of the processor model.
- **ProcessorPhysicalCores** Number of physical cores in the processor.
- **ProcessorUpdateRevision** The microcode version.
- **SocketCount** Number of physical CPU sockets of the machine.
- **SpeculationControl** If the system has enabled protections needed to validate the speculation control vulnerability.

### Census.Security

This event provides information on about security settings used to help keep Windows up-to-date and secure.

- **AvailableSecurityProperties** Enumerates and reports state on the relevant security properties for Device

Guard.

- **CGRunning** Is Credential Guard running?
- **DGState** A summary of the Device Guard state.
- **HVCIRunning** Is HVCI running?
- **IsSawGuest** Describes whether the device is running as a Secure Admin Workstation Guest.
- **IsSawHost** Describes whether the device is running as a Secure Admin Workstation Host.
- **RequiredSecurityProperties** Describes the required security properties to enable virtualization-based security.
- **SecureBootCapable** Is this device capable of running Secure Boot?
- **VBSState** Is virtualization-based security enabled, disabled, or running?

### Census.Speech

This event is used to gather basic speech settings on the device.

The following fields are available:

- **AboveLockEnabled** Cortana setting that represents if Cortana can be invoked when the device is locked.
- **GPAllowInputPersonalization** Indicates if a Group Policy setting has enabled speech functionalities.
- **HolographicSpeechInputDisabled** Holographic setting that represents if the attached HMD devices have speech functionality disabled by the user.
- **HolographicSpeechInputDisabledRemote** Indicates if a remote policy has disabled speech functionalities for the HMD devices.
- **KWSEnabled** "Cortana setting that represents if a user has enabled the ""Hey Cortana"" keyword spotter (KWS)."
- **MDMAllowInputPersonalization** Indicates if an MDM policy has enabled speech functionalities.
- **RemotelyManaged** Indicates if the device is being controlled by a remote administrator (MDM or Group Policy) in the context of speech functionalities.
- **SpeakerIdEnabled** Cortana setting that represents if keyword detection has been trained to try to respond to a single user's voice.
- **SpeechServicesEnabled** Windows setting that represents whether a user is opted-in for speech services on the device.

### Census.Storage

This event sends data about the total capacity of the system volume and primary disk, to help keep Windows up to date.

The following fields are available:

- **PrimaryDiskTotalCapacity** Retrieves the amount of disk space on the primary disk of the device in MB.
- **PrimaryDiskType** Retrieves an enumerator value of type STORAGE\_BUS\_TYPE that indicates the type of bus to which the device is connected. This should be used to interpret the raw device properties at the end of this structure (if any).
- **SystemVolumeTotalCapacity** Retrieves the size of the partition that the System volume is installed on in MB.

### Census.Userdefault

This event sends data about the current user's default preferences for browser and several of the most popular extensions and protocols, to help keep Windows up to date.

The following fields are available:

- **DefaultApp** The current user's default program selected for the following extension or protocol: .html,.htm,.jpg,.jpeg,.png,.mp3,.mp4, .mov,.pdf
- **DefaultBrowserProgId** The ProgramId of the current user's default browser

## Census.UserNLS

This event sends data about the default app language, input, and display language preferences set by the user, to help keep Windows up to date.

The following fields are available:

- **DefaultAppLanguage** The current user Default App Language.
- **DisplayLanguage** The current user preferred Windows Display Language.
- **HomeLocation** The current user location, which is populated using GetUserGeold() function.
- **KeyboardInputLanguages** The Keyboard input languages installed on the device.
- **SpeechInputLanguages** The Speech Input languages installed on the device.

## Census.VM

This event sends data indicating whether virtualization is enabled on the device, and its various characteristics, to help keep Windows up to date.

The following fields are available:

- **CloudService** Indicates which cloud service, if any, that this virtual machine is running within.
- **HyperVisor** Retrieves whether the current OS is running on top of a Hypervisor.
- **IOMMUPresent** Represents if an input/output memory management unit (IOMMU) is present.
- **isVDI** Is the device using Virtual Desktop Infrastructure?
- **IsVirtualDevice** Retrieves that when the Hypervisor is Microsoft's Hyper-V Hypervisor or other Hv#HASH#1 Hypervisor, this field will be set to FALSE for the Hyper-V host OS and TRUE for any guest OS's. This field should not be relied upon for non-Hv#HASH#1 Hypervisors.
- **SLATSupported** Represents whether Second Level Address Translation (SLAT) is supported by the hardware.
- **VirtualizationFirmwareEnabled** Represents whether virtualization is enabled in the firmware.

## Census.WU

This event sends data about the Windows update server and other App store policies, to help keep Windows up to date.

The following fields are available:

- **AppraiserGatedStatus** Indicates whether a device has been gated for upgrading.
- **AppStoreAutoUpdate** Retrieves the Appstore settings for auto upgrade. (Enable/Disabled).
- **AppStoreAutoUpdateMDM** Retrieves the App Auto Update value for MDM: 0 - Disallowed. 1 - Allowed. 2 - Not configured. Default: [2] Not configured
- **AppStoreAutoUpdatePolicy** Retrieves the Microsoft Store App Auto Update group policy setting
- **DelayUpgrade** Retrieves the Windows upgrade flag for delaying upgrades.
- **OSAssessmentFeatureOutOfDate** How many days has it been since a the last feature update was released but the device did not install it?
- **OSAssessmentForFeatureUpdate** Is the device is on the latest feature update?
- **OSAssessmentForQualityUpdate** Is the device on the latest quality update?
- **OSAssessmentForSecurityUpdate** Is the device on the latest security update?
- **OSAssessmentQualityOutOfDate** How many days has it been since a the last quality update was released but the device did not install it?
- **OSAssessmentReleaseInfoTime** The freshness of release information used to perform an assessment.
- **OSRollbackCount** The number of times feature updates have rolled back on the device.
- **OSRolledBack** A flag that represents when a feature update has rolled back during setup.
- **OSUninstalled** A flag that represents when a feature update is uninstalled on a device .
- **OSWUAutoUpdateOptions** Retrieves the auto update settings on the device.



- **UninstallActive** A flag that represents when a device has uninstalled a previous upgrade recently.
- **UpdateServiceURLConfigured** Retrieves if the device is managed by Windows Server Update Services (WSUS).
- **WUDeferUpdatePeriod** Retrieves if deferral is set for Updates
- **WUDeferUpgradePeriod** Retrieves if deferral is set for Upgrades
- **WUDODownloadMode** Retrieves whether DO is turned on and how to acquire/distribute updates Delivery Optimization (DO) allows users to deploy previously downloaded WU updates to other devices on the same network.
- **WUMachineId** Retrieves the Windows Update (WU) Machine Identifier.
- **WUPauseState** Retrieves WU setting to determine if updates are paused
- **WUServer** Retrieves the HTTP(S) URL of the WSUS server that is used by Automatic Updates and API callers (by default).

### Census.Xbox

This event sends data about the Xbox Console, such as Serial Number and DeviceId, to help keep Windows up to date.

The following fields are available:

- **XboxConsolePreferredLanguage** Retrieves the preferred language selected by the user on Xbox console.
- **XboxConsoleSerialNumber** Retrieves the serial number of the Xbox console.
- **XboxLiveDeviceId** Retrieves the unique device id of the console.
- **XboxLiveSandboxId** Retrieves the developer sandbox id if the device is internal to MS.

## Diagnostic data events

### TelClientSynthetic.AuthorizationInfo\_Startup

This event sends data indicating that a device has undergone a change of diagnostic data opt-in level detected at UTC startup, to help keep Windows up to date.

The following fields are available:

- **CanAddMsaToMsTelemetry** True if UTC is allowed to add MSA user identity onto diagnostic data from the OS provider groups.
- **CanCollectAnyTelemetry** True if UTC is allowed to collect non-OS diagnostic data. Non-OS diagnostic data is responsible for providing its own opt-in mechanism.
- **CanCollectCoreTelemetry** True if UTC is allowed to collect data which is tagged with both MICROSOFT\_KEYWORD\_CRITICAL\_DATA and MICROSOFT\_EVENTTAG\_CORE\_DATA.
- **CanCollectHeartbeats** True if UTC is allowed to collect heartbeats.
- **CanCollectOsTelemetry** True if UTC is allowed to collect diagnostic data from the OS provider groups.
- **CanPerformDiagnosticEscalations** True if UTC is allowed to perform all scenario escalations.
- **CanPerformScripting** True if UTC is allowed to perform scripting.
- **CanPerformTraceEscalations** True if UTC is allowed to perform scenario escalations with tracing actions.
- **CanReportScenarios** True if UTC is allowed to load and report scenario completion, failure, and cancellation events.
- **PreviousPermissions** Bitmask representing the previously configured permissions since the diagnostic data client was last started.
- **TransitionFromEverythingOff** True if this transition is moving from not allowing core diagnostic data to allowing core diagnostic data.

### TelClientSynthetic.AuthorizationInfo\_RuntimeTransition

This event sends data indicating that a device has undergone a change of diagnostic data opt-in level during the

runtime of the device (not at UTC boot or offline), to help keep Windows up to date.

The following fields are available:

- **CanAddMsaToMsTelemetry** True if UTC is allowed to add MSA user identity onto diagnostic data from the OS provider groups.
- **CanCollectAnyTelemetry** True if UTC is allowed to collect non-OS diagnostic data. Non-OS diagnostic data is responsible for providing its own opt-in mechanism.
- **CanCollectCoreTelemetry** True if UTC is allowed to collect data which is tagged with both MICROSOFT\_KEYWORD\_CRITICAL\_DATA and MICROSOFT\_EVENTTAG\_CORE\_DATA.
- **CanCollectHeartbeats** True if UTC is allowed to collect heartbeats.
- **CanCollectOsTelemetry** True if UTC is allowed to collect diagnostic data from the OS provider groups.
- **CanPerformDiagnosticEscalations** True if UTC is allowed to perform all scenario escalations.
- **CanPerformScripting** True if UTC is allowed to perform scripting.
- **CanPerformTraceEscalations** True if UTC is allowed to perform scenario escalations with tracing actions.
- **CanReportScenarios** True if UTC is allowed to load and report scenario completion, failure, and cancellation events.
- **PreviousPermissions** Bitmask representing the previously configured permissions since the diagnostic data opt-in level was last changed.
- **TransitionFromEverythingOff** True if this transition is moving from not allowing core diagnostic data to allowing core diagnostic data.

#### TelClientSynthetic.ConnectivityHeartBeat\_0

This event sends data about the connectivity status of the Connected User Experience and Telemetry component that uploads diagnostic data events. If an unrestricted free network (such as Wi-Fi) is available, this event updates the last successful upload time. Otherwise, it checks whether a Connectivity Heartbeat event was fired in the past 24 hours, and if not, it fires an event. A Connectivity Heartbeat event also fires when a device recovers from costed network to free network.

The following fields are available:

- **CensusExitCode** Returns last execution codes from census client run.
- **CensusStartTime** Returns timestamp corresponding to last successful census run.
- **CensusTaskEnabled** Returns Boolean value for the census task (Enable/Disable) on client machine.
- **LastConnectivityLossTime** Retrieves the last time the device lost free network.
- **LastConntectivityLossTime** Retrieves the last time the device lost free network.
- **NetworkState** Retrieves the network state: 0 = No network. 1 = Restricted network. 2 = Free network.
- **NoNetworkTime** Retrieves the time spent with no network (since the last time) in seconds.
- **RestrictedNetworkTime** Retrieves the time spent on a metered (cost restricted) network in seconds.

#### TelClientSynthetic.HeartBeat\_5

This event sends data about the health and quality of the diagnostic data from the given device, to help keep Windows up to date. It also enables data analysts to determine how 'trusted' the data is from a given device.

The following fields are available:

- **AgentConnectionErrorsCount** The number of non-timeout errors associated with the host/agent channel.
- **CensusExitCode** The last exit code of the Census task.
- **CensusStartTime** The time of the last Census run.
- **CensusTaskEnabled** Indicates whether Census is enabled.
- **ConsumerDroppedCount** The number of events dropped by the consumer layer of the diagnostic data client.
- **CriticalDataDbDroppedCount** The number of critical data sampled events that were dropped at the database

layer.

- **CriticalDataThrottleDroppedCount** The number of critical data sampled events that were dropped because of throttling.
- **CriticalOverflowEntersCounter** The number of times a critical overflow mode was entered into the event database.
- **DbCriticalDroppedCount** The total number of dropped critical events in the event database.
- **DbDroppedCount** The number of events that were dropped because the database was full.
- **DecodingDroppedCount** The number of events dropped because of decoding failures.
- **EnteringCriticalOverflowDroppedCounter** The number of events that was dropped because a critical overflow mode was initiated.
- **EtwDroppedBufferCount** The number of buffers dropped in the CUET ETW session.
- **EtwDroppedCount** The number of events dropped by the ETW layer of the diagnostic data client.
- **EventSubStoreResetCounter** The number of times the event database was reset.
- **EventSubStoreResetSizeSum** The total size of the event database across all resets reports in this instance.
- **EventsUploaded** The number of events that have been uploaded.
- **Flags** Flags that indicate device state, such as network, battery, and opt-in state.
- **FullTriggerBufferDroppedCount** The number of events that were dropped because the trigger buffer was full.
- **HeartBeatSequenceNumber** A monotonically increasing heartbeat counter.
- **InvalidHttpCodeCount** The number of invalid HTTP codes received from Vortex.
- **LastAgentConnectionError** The last non-timeout error that happened in the host/agent channel.
- **LastEventSizeOffender** The name of the last event that exceeded the maximum event size.
- **LastInvalidHttpCode** The last invalid HTTP code received from Vortex.
- **MaxActiveAgentConnectionCount** The maximum number of active agents during this heartbeat timeframe.
- **MaxInUseScenarioCounter** The soft maximum number of scenarios loaded by the Connected User Experiences and Telemetry component.
- **PreviousHeartBeatTime** The time of last heartbeat event. This allows chaining of events.
- **SettingsHttpAttempts** The number of attempts to contact the OneSettings service.
- **SettingsHttpFailures** The number of failures from contacting the OneSettings service.
- **ThrottledDroppedCount** The number of events dropped due to throttling of noisy providers.
- **UploaderDroppedCount** The number of events dropped by the uploader layer of the diagnostic data client.
- **VortexFailuresTimeout** The number of timeout failures received from Vortex.
- **VortexHttpAttempts** The number of attempts to contact the Vortex service.
- **VortexHttpFailures4xx** The number of 400-499 error codes received from Vortex.
- **VortexHttpFailures5xx** The number of 500-599 error codes received from Vortex.

#### **TelClientSynthetic.PrivacySettingsAfterCreatorsUpdate**

This event sends basic data on privacy settings before and after a feature update. This is used to ensure that customer privacy settings are correctly migrated across feature updates.

The following fields are available:

- **PostUpgradeSettings** The privacy settings after a feature update.
- **PreUpgradeSettings** The privacy settings before a feature update.

## DxgKernelTelemetry events

#### **DxgKrnTelemetry.GPUAdapterInventoryV2**

This event sends basic GPU and display driver information to keep Windows and display drivers up-to-date.

The following fields are available:

- **aiSeqId** The event sequence ID.
- **bootId** The system boot ID.
- **ComputePreemptionLevel** The maximum preemption level supported by GPU for compute payload.
- **DedicatedSystemMemoryB** The amount of system memory dedicated for GPU use (in bytes).
- **DedicatedVideoMemoryB** The amount of dedicated VRAM of the GPU (in bytes).
- **DisplayAdapterLuid** The display adapter LUID.
- **DriverDate** The date of the display driver.
- **DriverRank** The rank of the display driver.
- **DriverVersion** The display driver version.
- **GPUDeviceID** The GPU device ID.
- **GPUPreemptionLevel** The maximum preemption level supported by GPU for graphics payload.
- **GPURevisionID** The GPU revision ID.
- **GPUVendorID** The GPU vendor ID.
- **InterfaceID** The GPU interface ID.
- **IsDisplayDevice** Does the GPU have displaying capabilities?
- **IsHybridDiscrete** Does the GPU have discrete GPU capabilities in a hybrid device?
- **IsHybridIntegrated** Does the GPU have integrated GPU capabilities in a hybrid device?
- **IsLDA** Is the GPU comprised of Linked Display Adapters?
- **IsMiracastSupported** Does the GPU support Miracast?
- **IsMismatchLDA** Is at least one device in the Linked Display Adapters chain from a different vendor?
- **IsMPOSsupported** Does the GPU support Multi-Plane Overlays?
- **IsMsMiracastSupported** Are the GPU Miracast capabilities driven by a Microsoft solution?
- **IsPostAdapter** Is this GPU the POST GPU in the device?
- **IsRenderDevice** Does the GPU have rendering capabilities?
- **IsSoftwareDevice** Is this a software implementation of the GPU?
- **MeasureEnabled** Is the device listening to MICROSOFT\_KEYWORD\_MEASURES?
- **SharedSystemMemoryB** The amount of system memory shared by GPU and CPU (in bytes).
- **SubSystemID** The subsystem ID.
- **SubVendorID** The GPU sub vendor ID.
- **TelemetryEnabled** Is the device listening to MICROSOFT\_KEYWORD\_TELEMETRY?
- **TellInvEvtTrigger** What triggered this event to be logged? Example: 0 (GPU enumeration) or 1 (DxgKrnlTelemetry provider toggling)
- **version** The event version.
- **WDDMVersion** The Windows Display Driver Model version.
- **NumVidPnSources** The number of supported display output sources.
- **NumVidPnTargets** The number of supported display output targets.

## Fault Reporting events

### Microsoft.Windows.FaultReporting.AppCrashEvent

"This event sends data about crashes for both native and managed applications, to help keep Windows up to date. The data includes information about the crashing process and a summary of its exception record. It does not contain any Watson bucketing information. The bucketing information is recorded in a Windows Error Reporting (WER) event that is generated when the WER client reports the crash to the Watson service, and the WER event will contain the same ReportID (see field 14 of crash event, field 19 of WER event) as the crash event for the crash being reported. AppCrash is emitted once for each crash handled by WER (e.g. from an unhandled exception or

FailFast or ReportException). Note that Generic Watson event types (e.g. from PLM) that may be considered crashes"" by a user DO NOT emit this event."

The following fields are available:

- **AppName** The name of the app that has crashed.
- **AppSessionGuid** GUID made up of process ID and is used as a correlation vector for process instances in the diagnostic data backend.
- **AppTimeStamp** The date/time stamp of the app.
- **AppVersion** The version of the app that has crashed.
- **ExceptionCode** The exception code returned by the process that has crashed.
- **ExceptionOffset** The address where the exception had occurred.
- **Flags** "Flags indicating how reporting is done. For example, queue the report, do not offer JIT debugging, or do not terminate the process after reporting. "
- **ModName** Exception module name (e.g. bar.dll).
- **ModTimeStamp** The date/time stamp of the module.
- **ModVersion** The version of the module that has crashed.
- **PackageFullName** Store application identity.
- **PackageRelativeAppId** Store application identity.
- **ProcessArchitecture** Architecture of the crashing process, as one of the PROCESSOR\_ARCHITECTURE\_\* constants: 0: PROCESSOR\_ARCHITECTURE\_INTEL. 5: PROCESSOR\_ARCHITECTURE\_ARM. 9: PROCESSOR\_ARCHITECTURE\_AMD64. 12: PROCESSOR\_ARCHITECTURE\_ARM64.
- **ProcessCreateTime** The time of creation of the process that has crashed.
- **ProcessId** The ID of the process that has crashed.
- **ReportId** A GUID used to identify the report. This can be used to track the report across Watson.
- **TargetAppId** The kernel reported AppId of the application being reported.
- **TargetAppVer** The specific version of the application being reported
- **TargetAsId** The sequence number for the hanging process.

## Feature update events

### **Microsoft.Windows.Upgrade.Uninstall.UninstallFailed**

This event sends diagnostic data about failures when uninstalling a feature update, to help resolve any issues preventing customers from reverting to a known state

The following fields are available:

- **failureReason** Provides data about the uninstall initialization operation failure
- **hr** Provides the Win32 error code for the operation failure

### **Microsoft.Windows.Upgrade.Uninstall.UninstallFinalizedAndRebootTriggered**

Indicates that the uninstall was properly configured and that a system reboot was initiated

The following fields are available:

- **name** Name of the event

## Hang Reporting events

### **Microsoft.Windows.HangReporting.AppHangEvent**

This event sends data about hangs for both native and managed applications, to help keep Windows up to date. It does not contain any Watson bucketing information. The bucketing information is recorded in a Windows Error

Reporting (WER) event that is generated when the WER client reports the hang to the Watson service, and the WER event will contain the same ReportID (see field 13 of hang event, field 19 of WER event) as the hang event for the hang being reported. AppHang is reported only on PC devices. It handles classic Win32 hangs and is emitted only once per report. Some behaviors that may be perceived by a user as a hang are reported by app managers (e.g. PLM/RM/EM) as Watson Generics and will not produce AppHang events.

The following fields are available:

- **AppName** The name of the app that has hung.
- **AppSessionGuid** GUID made up of process id used as a correlation vector for process instances in the diagnostic data backend.
- **AppVersion** The version of the app that has hung.
- **PackageFullName** Store application identity.
- **PackageRelativeAppId** Store application identity.
- **ProcessArchitecture** Architecture of the hung process, as one of the PROCESSOR\_ARCHITECTURE\_\* constants: 0: PROCESSOR\_ARCHITECTURE\_INTEL. 5: PROCESSOR\_ARCHITECTURE\_ARM. 9: PROCESSOR\_ARCHITECTURE\_AMD64. 12: PROCESSOR\_ARCHITECTURE\_ARM64.
- **ProcessCreateTime** The time of creation of the process that has hung.
- **ProcessId** The ID of the process that has hung.
- **ReportId** A GUID used to identify the report. This can be used to track the report across Watson.
- **TargetAppId** The kernel reported AppId of the application being reported.
- **TargetAppVer** The specific version of the application being reported.
- **TargetAsId** The sequence number for the hanging process.
- **TypeCode** Bitmap describing the hang type.
- **WaitingOnAppName** If this is a cross process hang waiting for an application, this has the name of the application.
- **WaitingOnAppVersion** If this is a cross process hang, this has the version of the application for which it is waiting.
- **WaitingOnPackageFullName** If this is a cross process hang waiting for a package, this has the full name of the package for which it is waiting.
- **WaitingOnPackageRelativeAppId** If this is a cross process hang waiting for a package, this has the relative application id of the package.

## Inventory events

### **Microsoft.Windows.Inventory.Core.InventoryDeviceUsbHubClassStartSync**

This event indicates that a new set of InventoryDeviceUsbHubClassAdd events will be sent

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events

### **Microsoft.Windows.Inventory.Core.InventoryDeviceUsbHubClassAdd**

This event sends basic metadata about the USB hubs on the device

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events
- **TotalUserConnectablePorts** Total number of connectable USB ports
- **TotalUserConnectableTypeCPorts** Total number of connectable USB Type C ports

### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeVBARuleViolationsAdd**

This event provides data on Microsoft Office VBA rule violations, including a rollup count per violation type, giving

an indication of remediation requirements for an organization. The event identifier is a unique GUID, associated with the validation rule

The following fields are available:

- **Count** Count of total Microsoft Office VBA rule violations

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeAddInAdd**

This event provides data on the installed Office Add-ins.

- **AddInCLSID** The CLSID key office for the Office addin.
- **AddInId** The identifier of the Office addin.
- **AddinType** The type of the Office addin.
- **BinFileTimestamp** The timestamp of the Office addin.
- **BinFileVersion** The version of the Office addin.
- **Description** The description of the Office addin.
- **FileId** The file ID of the Office addin.
- **FriendlyName** The friendly name of the Office addin.
- **FullPath** The full path to the Office addin.
- **LoadBehavior** A UInt32 that describes the load behavior.
- **LoadTime** The load time for the Office addin.
- **OfficeApplication** The Office application for this addin.
- **OfficeArchitecture** The architecture of the addin.
- **OfficeVersion** The Office version for this addin.
- **OutlookCrashingAddin** A boolean value that indicates if crashes have been found for this addin.
- **ProductCompany** The name of the company associated with the Office addin.
- **ProductName** The product name associated with the Office addin.
- **ProductVersion** The version associated with the Office addin.
- **ProgramId** The unique program identifier of the Office addin.
- **Provider** The provider name for this addin.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeAddInRemove**

This event indicates that the particular data object represented by the objectInstanceId is no longer present.

There are no fields in this event.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeInsightsAdd**

This event provides insight data on the installed Office products.

The following fields are available:

- **OfficeApplication** The name of the Office application.
- **OfficeArchitecture** The bitness of the Office application.
- **OfficeVersion** The version of the Office application.
- **Value** The insights collected about this entity.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeInsightsRemove**

This event indicates that the particular data object represented by the objectInstanceId is no longer present.

There are no fields in this event.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeInsightsStartSync**

This diagnostic event indicates that a new sync is being generated for this object type.

There are no fields in this event.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeSettingsAdd**

This event describes various Office settings.

The following fields are available:

- **BrowserFlags** Browser flags for Office-related products.
- **ExchangeProviderFlags** Provider policies for Office Exchange.
- **SharedComputerLicensing** Office shared computer licensing policies.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeSettingsStartSync**

Diagnostic event to indicate a new sync is being generated for this object type.

There are no fields in this event.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeVBAAdd**

This event provides a summary rollup count of conditions encountered while performing a local scan of Office files, analyzing for known VBA programmability compatibility issues between legacy office version and ProPlus, and between 32 and 64-bit versions

The following fields are available:

- **Design** Count of files with design issues found
- **Design\_x64** Count of files with 64 bit design issues found
- **DuplicateVBA** Count of files with duplicate VBA code
- **HasVBA** Count of files with VBA code
- **Inaccessible** Count of files that were inaccessible for scanning
- **Issues** Count of files with issues detected
- **Issues\_x64** Count of files with 64-bit issues detected
- **IssuesNone** Count of files with no issues detected
- **IssuesNone\_x64** Count of files with no 64-bit issues detected
- **Locked** Count of files that were locked, preventing scanning
- **NoVBA** Count of files with no VBA inside
- **Protected** Count of files that were password protected, preventing scanning
- **RemLimited** Count of files that require limited remediation changes
- **RemLimited\_x64** Count of files that require limited remediation changes for 64-bit issues
- **RemSignificant** Count of files that require significant remediation changes
- **RemSignificant\_x64** Count of files that require significant remediation changes for 64-bit issues
- **Score** Overall compatibility score calculated for scanned content
- **Score\_x64** Overall 64-bit compatibility score calculated for scanned content
- **Total** Total number of files scanned
- **Validation** Count of files that require additional manual validation
- **Validation\_x64** Count of files that require additional manual validation for 64-bit issues

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeVBARemove**

This event indicates that the particular data object represented by the objectInstanceId is no longer present.

There are no fields in this event.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeVBARuleViolationsRemove**

This event indicates that the particular data object represented by the objectInstanceId is no longer present.

There are no fields in this event.



### **Microsoft.Windows.Inventory.Core.InventoryApplicationFrameworkStartSync**

This event indicates that a new set of InventoryApplicationFrameworkAdd events will be sent

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events

### **Microsoft.Windows.Inventory.Core.InventoryApplicationFrameworkAdd**

This event provides the basic metadata about the frameworks an application may depend on

The following fields are available:

- **FileId** A hash that uniquely identifies a file
- **Frameworks** The list of frameworks this file depends on
- **InventoryVersion** The version of the inventory file generating the events
- **ProgramId** A hash of the Name, Version, Publisher, and Language of an application used to identify it

### **Microsoft.Windows.Inventory.Indicators.InventoryMiscellaneousUexIndicatorAdd**

These events represent the basic metadata about the OS indicators installed on the system which are used for keeping the device up-to-date.

The following fields are available:

- **IndicatorValue** The indicator value
- **Value** Describes an operating system indicator that may be relevant for the device upgrade.

### **Microsoft.Windows.Inventory.Indicators.Checksum**

This event summarizes the counts for the InventoryMiscellaneousUexIndicatorAdd events.

The following fields are available:

- **ChecksumDictionary** A count of each operating system indicator.
- **PCFP** Equivalent to the InventoryId field that is found in other core events.

### **Microsoft.Windows.Inventory.Core.InventoryApplicationAdd**

This event sends basic metadata about an application on the system to help keep Windows up to date.

The following fields are available:

- **HiddenArp** Indicates whether a program hides itself from showing up in ARP.
- **InstallDate** The date the application was installed (a best guess based on folder creation date heuristics).
- **InstallDateArpLastModified** The date of the registry ARP key for a given application. Hints at install date but not always accurate. Passed as an array. Example: 4/11/2015 00:00:00
- **InstallDateFromLinkFile** The estimated date of install based on the links to the files. Passed as an array.
- **InstallDateMsi** The install date if the application was installed via MSI. Passed as an array.
- **InventoryVersion** The version of the inventory file generating the events.
- **Language** The language code of the program.
- **MsiPackageCode** A GUID that describes the MSI Package. Multiple 'Products' (apps) can make up an MsiPackage.
- **MsiProductCode** A GUID that describe the MSI Product.
- **Name** The name of the application
- **OSVersionAtInstallTime** The four octets from the OS version at the time of the application's install.
- **PackageFullName** The package full name for a Store application.
- **ProgramInstanceId** A hash of the file IDs in an app.

- **Publisher** The Publisher of the application. Location pulled from depends on the 'Source' field.
- **RootDirPath** The path to the root directory where the program was installed.
- **Source** How the program was installed (ARP, MSI, Appx, etc...)
- **StoreAppType** A sub-classification for the type of Microsoft Store app, such as UWP or Win8StoreApp.
- **Type** "One of (""Application"", ""Hotfix"", ""BOE"", ""Service"", ""Unknown""). Application indicates Win32 or Appx app, Hotfix indicates app updates (KBs), BOE indicates it's an app with no ARP or MSI entry, Service indicates that it is a service. Application and BOE are the ones most likely seen."
- **Version** The version number of the program.

#### **Microsoft.Windows.Inventory.Core.InventoryApplicationRemove**

This event indicates that a new set of InventoryDevicePnpAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryApplicationStartSync**

This event indicates that a new set of InventoryApplicationAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDeviceContainerRemove**

This event indicates that the InventoryDeviceContainer object is no longer present.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDriverPackageAdd**

This event sends basic metadata about drive packages installed on the system to help keep Windows up-to-date.

The following fields are available:

- **Class** The class name for the device driver.
- **ClassGuid** The class GUID for the device driver.
- **Date** The driver package date.
- **Directory** The path to the driver package.
- **DriverInBox** Is the driver included with the operating system?
- **Inf** The INF name of the driver package.
- **InventoryVersion** The version of the inventory file generating the events.
- **Provider** The provider for the driver package.
- **SubmissionId** The HLK submission ID for the driver package.
- **Version** The version of the driver package.

#### **Microsoft.Windows.Inventory.Core.InventoryDriverBinaryStartSync**

This event indicates that a new set of InventoryDriverBinaryAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDriverBinaryRemove**

This event indicates that the InventoryDriverBinary object is no longer present.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDriverPackageRemove**

This event indicates that the InventoryDriverPackageRemove object is no longer present.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDevicePnpRemove**

This event indicates that the InventoryDevicePnpRemove object is no longer present.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDeviceContainerAdd**

This event sends basic metadata about a device container (such as a monitor or printer as opposed to a PNP device) to help keep Windows up-to-date.

The following fields are available:

- **Categories** A comma separated list of functional categories in which the container belongs.
- **DiscoveryMethod** The discovery method for the device container.
- **FriendlyName** The name of the device container.
- **InventoryVersion** The version of the inventory file generating the events.
- **IsActive** Is the device connected, or has it been seen in the last 14 days?
- **IsConnected** For a physically attached device, this value is the same as IsPresent. For wireless a device, this value represents a communication link.
- **IsMachineContainer** Is the container the root device itself?
- **IsNetworked** Is this a networked device?
- **IsPaired** Does the device container require pairing?
- **Manufacturer** The manufacturer name for the device container.
- **ModelId** A model GUID.
- **ModelName** The model name.
- **ModelNumber** The model number for the device container.
- **PrimaryCategory** The primary category for the device container.

#### **Microsoft.Windows.Inventory.Core.InventoryDeviceContainerStartSync**

This event indicates that a new set of InventoryDeviceContainerAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDeviceMediaClassStartSync**

This event indicates that a new set of InventoryDeviceMediaClassSAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDriverPackageStartSync**

This event indicates that a new set of InventoryDriverPackageAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDeviceMediaClassRemove**

This event indicates that the InventoryDeviceMediaClassRemove object is no longer present.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDevicePnpStartSync**

This event indicates that a new set of InventoryDevicePnpAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDeviceMediaClassAdd**

This event sends additional metadata about a PNP device that is specific to a particular class of devices to help keep Windows up to date while reducing overall size of data payload.

The following fields are available:

- **Audio\_CaptureDriver** The Audio device capture driver endpoint.
- **Audio\_RenderDriver** The Audio device render driver endpoint.
- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDevicePnpAdd**

This event represents the basic metadata about a PNP device and its associated driver

The following fields are available:

- **class** The device setup class of the driver loaded for the device
- **classGuid** The device class GUID from the driver package
- **COMPID** A JSON array the provides the value and order of the compatible ID tree for the device.
- **ContainerId** A system-supplied GUID that uniquely groups the functional devices associated with a single-function or multifunction device installed in the device.
- **description** The device description
- **deviceState** DeviceState is a bitmask of the following: DEVICE\_IS\_CONNECTED 0x0001 (currently only for container). DEVICE\_IS\_NETWORK\_DEVICE 0x0002 (currently only for container). DEVICE\_IS\_PAIRED 0x0004 (currently only for container). DEVICE\_IS\_ACTIVE 0x0008 (currently never set). DEVICE\_IS\_MACHINE 0x0010 (currently only for container). DEVICE\_IS\_PRESENT 0x0020 (currently always set). DEVICE\_IS\_HIDDEN 0x0040. DEVICE\_IS\_PRINTER 0x0080 (currently only for container). DEVICE\_IS\_WIRELESS 0x0100. DEVICE\_IS\_WIRELESS\_FAT 0x0200. The most common values are therefore: 32 (0x20)= device is present. 96 (0x60)= device is present but hidden. 288 (0x120)= device is a wireless device that is present
- **DriverId** A unique identifier for the installed device.
- **DriverName** The name of the driver image file.
- **driverPackageStrongName** The immediate parent directory name in the Directory field of InventoryDriverPackage.
- **driverVerDate** The date of the driver loaded for the device
- **driverVerVersion** The version of the driver loaded for the device
- **enumerator** The bus that enumerated the device
- **HWID** A JSON array that provides the value and order of the HWID tree for the device.
- **Inf** The INF file name.
- **installState** The device installation state. One of these values:

<https://msdn.microsoft.com/library/windows/hardware/ff543130.aspx>

- **InventoryVersion** The version of the inventory file generating the events.
- **lowerClassFilters** Lower filter class drivers IDs installed for the device.
- **lowerFilters** Lower filter drivers IDs installed for the device
- **manufacturer** The device manufacturer
- **matchingID** Represents the hardware ID or compatible ID that Windows uses to install a device instance
- **model** The device model
- **parentId** Device instance id of the parent of the device
- **ProblemCode** The current error code for the device.
- **provider** The device provider
- **service** The device service name#N##N##N##N##N#
- **STACKID** A JSON array that provides the value and order of the STACKID tree for the device.
- **upperClassFilters** Upper filter class drivers IDs installed for the device
- **upperFilters** Upper filter drivers IDs installed for the device

### **Microsoft.Windows.Inventory.Core.InventoryDriverBinaryAdd**

This event provides the basic metadata about driver binaries running on the system

The following fields are available:

- **DriverChecksum** The checksum of the driver file.
- **DriverCompany** The company name that developed the driver.
- **driverInBox** Is the driver included with the operating system?
- **driverIsKernelMode** Is it a kernel mode driver?
- **DriverName** The file name of the driver.
- **driverPackageStrongName** The strong name of the driver package
- **driverSigned** The strong name of the driver package
- **DriverTimeStamp** The low 32 bits of the time stamp of the driver file.
- **DriverType** A bitfield of driver attributes: 1. define DRIVER\_MAP\_DRIVER\_TYPE\_PRINTER 0x0001. 2. define DRIVER\_MAP\_DRIVER\_TYPE\_KERNEL 0x0002. 3. define DRIVER\_MAP\_DRIVER\_TYPE\_USER 0x0004. 4. define DRIVER\_MAP\_DRIVER\_IS\_SIGNED 0x0008. 5. define DRIVER\_MAP\_DRIVER\_IS\_INBOX 0x0010. 6. define DRIVER\_MAP\_DRIVER\_IS\_WINQUAL 0x0040. 7. define DRIVER\_MAP\_DRIVER\_IS\_SELF\_SIGNED 0x0020. 8. define DRIVER\_MAP\_DRIVER\_IS\_CI\_SIGNED 0x0080. 9. define DRIVER\_MAP\_DRIVER\_HAS\_BOOT\_SERVICE 0x0100. 10. define DRIVER\_MAP\_DRIVER\_TYPE\_I386 0x10000. 11. define DRIVER\_MAP\_DRIVER\_TYPE\_IA64 0x20000. 12. define DRIVER\_MAP\_DRIVER\_TYPE\_AMD64 0x40000. 13. define DRIVER\_MAP\_DRIVER\_TYPE\_ARM 0x100000. 14. define DRIVER\_MAP\_DRIVER\_TYPE\_THUMB 0x200000. 15. define DRIVER\_MAP\_DRIVER\_TYPE\_ARMNT 0x400000. 16. define DRIVER\_MAP\_DRIVER\_IS\_TIME\_STAMPED 0x800000.
- **DriverVersion** The version of the driver file.
- **ImageSize** The size of the driver file.
- **Inf** The name of the INF file.
- **InventoryVersion** The version of the inventory file generating the events.
- **Product** The product name that is included in the driver file.
- **ProductVersion** The product version that is included in the driver file.
- **service** The device service name
- **WdfVersion** The Windows Driver Framework version.

### **Microsoft.Windows.Inventory.Indicators.InventoryMiscellaneousUexIndicator**

This event sends value data about the markers on custom devices, to help keep Windows up to date. The formal

name for markers is UEX Indicators. See marker list for definitions.

The following fields are available:

- **IndicatorValue** Value of the marker/indicator
- **Key** Name of the marker/indicator

#### **Microsoft.Windows.Inventory.Core.AmiTelCacheVersions**

This event sends inventory component versions for the Device Inventory data.

The following fields are available:

- **aeinv** The version of the App inventory component.
- **devinv** The file version of the Device inventory component.

#### **Microsoft.Windows.Inventory.Core.AmiTelCacheChecksum**

This event captures basic checksum data about the device inventory items stored in the cache for use in validating data completeness for Microsoft.Windows.Inventory.Core events. The fields in this event may change over time, but they will always represent a count of a given object.

The following fields are available:

- **Device** A count of device objects in cache
- **DeviceCensus** A count of devicecensus objects in cache
- **DriverPackageExtended** A count of driverpackageextended objects in cache
- **File** A count of file objects in cache
- **FileSigningInfo** A count of file signing info objects in cache.
- **Generic** A count of generic objects in cache
- **HwItem** A count of hwitem objects in cache
- **InventoryApplication** A count of application objects in cache
- **InventoryApplicationFile** A count of application file objects in cache
- **InventoryDeviceContainer** A count of device container objects in cache
- **InventoryDeviceInterface** A count of inventory device interface objects in cache.
- **InventoryDeviceMediaClass** A count of device media objects in cache
- **InventoryDevicePnp** A count of devicepnp objects in cache
- **InventoryDriverBinary** A count of driver binary objects in cache
- **InventoryDriverPackage** A count of device objects in cache
- **Metadata** A count of metadata objects in cache
- **Orphan** A count of orphan file objects in cache
- **Programs** A count of program objects in cache

#### **Microsoft.Windows.Inventory.Core.InventoryDeviceInterfaceStartSync**

This event indicates that a new set of InventoryDeviceInterfaceAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDeviceInterfaceAdd**

This event retrieves information about what sensor interfaces are available on the device.

The following fields are available:

- **Accelerometer3D** Indicates if an Accelerator3D sensor is found.
- **ActivityDetection** Indicates if an Activity Detection sensor is found.

- **AmbientLight** Indicates if an Ambient Light sensor is found.
- **Barometer** Indicates if a Barometer sensor is found.
- **Custom** Indicates if a Custom sensor is found.
- **EnergyMeter** Indicates if an Energy sensor is found.
- **FloorElevation** Indicates if a Floor Elevation sensor is found.
- **GeomagneticOrientation** Indicates if a Geo Magnetic Orientation sensor is found.
- **GravityVector** Indicates if a Gravity Detector sensor is found.
- **Gyrometer3D** Indicates if a Gyrometer3D sensor is found.
- **Humidity** Indicates if a Humidity sensor is found.
- **InventoryVersion** The version of the inventory file generating the events.
- **LinearAccelerometer** Indicates if a Linear Accelerometer sensor is found.
- **Magnetometer3D** Indicates if a Magnetometer3D sensor is found.
- **Orientation** Indicates if an Orientation sensor is found.
- **Pedometer** Indicates if a Pedometer sensor is found.
- **Proximity** Indicates if a Proximity sensor is found.
- **RelativeOrientation** Indicates if a Relative Orientation sensor is found.
- **SimpleDeviceOrientation** Indicates if a Simple Device Orientation sensor is found.
- **Temperature** Indicates if a Temperature sensor is found.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeAddInStartSync**

This event indicates that a new sync is being generated for this object type.

There are no fields in this event.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeIdentifiersAdd**

This event provides data on the installed Office identifiers.

- **OAudienceData** The Office Audience descriptor.
- **OAudienceId** The Office Audience ID.
- **OMID** The Office machine ID.
- **OPlatform** The Office architecture.
- **OVersion** The Office version
- **OTenantId** The Office 365 Tenant GUID.
- **OWowMID** The Office machine ID.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeIdentifiersStartSync**

This event indicates that a new sync is being generated for this object type.

There are no fields in this event.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeIESettingsAdd**

This event provides data on the installed Office-related Internet Explorer features.

- **OleFeatureAddon** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OleMachineLockdown** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OleMimeHandling** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OleMimeSniffing** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OleNoAxInstall** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OleNoDownload** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OleObjectCaching** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OlePasswordDisable** For more information, see the Office-related [Internet Feature Control Keys](#).

- **OleSafeBind** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OleSecurityBand** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OleUncSaveCheck** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OleValidateUrl** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OleWebOcPopup** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OleWinRestrict** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OleZoneElevate** For more information, see the Office-related [Internet Feature Control Keys](#).

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeIETSettingsStartSync**

This event indicates that a new sync is being generated for this object type.

There are no fields in this event.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeProductsAdd**

This event describes the Office products that are installed.

- **OC2rApps** The Office Click-to-Run apps.
- **OC2rSkus** The Office Click-to-Run products.
- **OMsiApps** The Office MSI apps.
- **OProductCodes** The Office MSI product code.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeProductsStartSync**

This event indicates that a new sync is being generated for this object type.

There are no fields in this event.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeVBARuleViolationsStartSync**

This event indicates that a new sync is being generated for this object type.

There are no fields in this event.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeVBAStartSync**

This event indicates that a new sync is being generated for this object type.

There are no fields in this event.

#### **Microsoft.Windows.Inventory.Indicators.InventoryMiscellaneousUexIndicatorRemove**

This event is a counterpart to `InventoryMiscellaneousUexIndicatorAdd` that indicates that the item has been removed.

There are no fields in this event.

#### **Microsoft.Windows.Inventory.Indicators.InventoryMiscellaneousUexIndicatorStartSync**

This event indicates that a new set of `InventoryMiscellaneousUexIndicatorAdd` events will be sent.

There are no fields in this event.

## OneDrive events

#### **Microsoft.OneDrive.Sync.Updater.OfficeRegistration**

This event determines the status of the OneDrive integration with Microsoft Office.

The following fields are available:

- **isValid** Is the Microsoft Office registration valid?

#### **Microsoft.OneDrive.Sync.Updater.UpdateTierReg**



This event determines status of the update tier registry values.

The following fields are available:

- **regReadEnterpriseHr** The HRESULT of the enterprise reg read value.
- **regReadTeamHr** The HRESULT of the team reg read value.

#### **Microsoft.OneDrive.Sync.Updater.RepairResult**

The event determines the result of the installation repair.

The following fields are available:

- **hr** The HRESULT of the operation.

#### **Microsoft.OneDrive.Sync.Updater.UpdateXmlDownloadHRESULT**

This event determines the status when downloading the OneDrive update configuration file.

The following fields are available:

- **hr** The HRESULT of the operation.

#### **Microsoft.OneDrive.Sync.Updater.SetupBinaryDownloadHRESULT**

This event indicates the status when downloading the OneDrive setup file.

The following fields are available:

- **hr** The HRESULT of the operation.

#### **Microsoft.OneDrive.Sync.Updater.UpdateOverallResult**

This event determines the outcome of the operation.

The following fields are available:

- **hr** The HRESULT of the operation.
- **IsLoggingEnabled** Is logging enabled?
- **UpdaterVersion** The version of the updater.

#### **Microsoft.OneDrive.Sync.Updater.WebConnectionStatus**

This event determines the error code that was returned when verifying Internet connectivity.

The following fields are available:

- **winInetError** The HRESULT of the operation.

#### **Microsoft.OneDrive.Sync.Updater.OverlayIconStatus**

This event indicates if the OneDrive overlay icon is working correctly. 0 = healthy; 1 = can be fixed; 2 = broken

The following fields are available:

- **32bit** The status of the OneDrive overlay icon on a 32-bit operating system.
- **64bit** The status of the OneDrive overlay icon on a 64-bit operating system.
- **SixtyFourBit** The status of the OneDrive overlay icon on a 32-bit operating system.
- **ThirtyTwoBit** The status of the OneDrive overlay icon on a 64-bit operating system.

#### **Microsoft.OneDrive.Sync.Updater.ComponentInstallState**

This event determines the installation state of dependent OneDrive components.

The following fields are available:

- **ComponentName** The name of the dependent component.

- **isInstalled** Is the dependent component installed?

#### **Microsoft.OneDrive.Sync.Updater.CommonData**

This event contains basic OneDrive configuration data that helps to diagnose failures.

The following fields are available:

- **AppVersion** The version of the app.
- **BuildArch** Is the architecture x86 or x64?
- **Environment** Is the device on the production or int service?
- **IsMSFTInternal** Is this an internal Microsoft device?
- **MachineGuid** The CEIP machine ID.
- **Market** Which market is this in?
- **OfficeVersion** The version of Office that is installed.
- **OneDriveDeviceId** The OneDrive device ID.
- **OSDeviceName** Only if the device is internal to Microsoft, the device name.
- **OSUserName** Only if the device is internal to Microsoft, the user name.
- **UserGuid** A unique global user identifier.

#### **Microsoft.OneDrive.Sync.Setup.APIOperation**

This event includes basic data about install and uninstall OneDrive API operations.

The following fields are available:

- **APIName** The name of the API.
- **Duration** How long the operation took.
- **IsSuccess** Was the operation successful?
- **ResultCode** The result code.
- **ScenarioName** The name of the scenario.

#### **Microsoft.OneDrive.Sync.Setup.RegisterStandaloneUpdaterAPIOperation**

This event is related to registering or unregistering the OneDrive update task.

The following fields are available:

- **APIName** The name of the API.
- **IsSuccess** Was the operation successful?
- **RegisterNewTaskResult** The HRESULT of the RegisterNewTask operation.
- **ScenarioName** The name of the scenario.
- **UnregisterOldTaskResult** The HRESULT of the UnregisterOldTask operation.

#### **Microsoft.OneDrive.Sync.Setup.EndExperience**

This event includes a success or failure summary of the installation.

The following fields are available:

- **APIName** The name of the API.
- **HResult** Indicates the result code of the event
- **IsSuccess** Was the operation successful?
- **ScenarioName** The name of the scenario.

#### **Microsoft.OneDrive.Sync.Setup.OSUpgradeInstallationOperation**

This event is related to the OS version when the OS is upgraded with OneDrive installed.

The following fields are available:

- **CurrentOneDriveVersion** The current version of OneDrive.
- **CurrentOSBuildBranch** The current branch of the operating system.
- **CurrentOSBuildNumber** The current build number of the operating system.
- **CurrentOSVersion** The current version of the operating system.
- **HResult** The HRESULT of the operation.
- **SourceOSBuildBranch** The source branch of the operating system.
- **SourceOSBuildNumber** The source build number of the operating system.
- **SourceOSVersion** The source version of the operating system.

#### Microsoft.OneDrive.Sync.Setup.SetupCommonData

This event contains basic OneDrive configuration data that helps to diagnose failures.

The following fields are available:

- **AppVersion** The version of the app.
- **BuildArchitecture** Is the architecture x86 or x64?
- **Environment** Is the device on the production or int service?
- **MachineGuid** The CEIP machine ID.
- **Market** Which market is this in?
- **MSFTInternal** Is this an internal Microsoft device?
- **OfficeVersionString** The version of Office that is installed.
- **OSDeviceName** Only if the device is internal to Microsoft, the device name.
- **OSUserName** Only if the device is internal to Microsoft, the user name.
- **UserGuid** The CEIP user ID.

## Remediation events

### NOTE

Events from this provider are sent with the installation of KB4023057 and any subsequent Windows update. For details, see [this support article](#).

#### Microsoft.Windows.Remediation.Applicable

Reports whether a specific remediation to issues preventing security and quality updates is applicable based on detection.

The following fields are available:

- **CV** Correlation vector.
- **DetectedCondition** Boolean true if detect condition is true and perform action will be run.
- **GlobalEventCounter** Client side counter which indicates ordering of events sent by the remediation system.
- **PackageVersion** Current package version of Remediation.
- **PluginName** Name of the remediation plugin specified for each generic plugin event.
- **RemediationShellDeviceManaged** TRUE if the device is WSUS managed or Windows Updated is disabled.
- **RemediationShellDeviceNewOS** TRUE if the device has a recently installed OS.
- **RemediationShellDeviceSccm** TRUE if the device is SCCM managed.
- **RemediationShellDeviceZeroExhaust** TRUE if the device has opted out of Windows Updates completely.
- **Result** Result for detection or perform action phases of the remediation system.

#### Microsoft.Windows.Remediation.ChangePowerProfileDetection

Indicates whether the remediation system can put in a request to defer a system-initiated sleep to enable installation of security or quality updates.

The following fields are available:

- **ActionName** A descriptive name for the plugin action.
- **CurrentPowerPlanGUID** The ID of the current power plan configured on the device.
- **CV** Correlation vector.
- **GlobalEventCounter** Counter that indicates the ordering of events on the device.
- **PackageVersion** Current package version of remediation service.
- **RemediationBatteryPowerBatteryLevel** Integer between 0 and 100 indicating % battery power remaining (if not on battery, expect 0).
- **RemediationFUInProgress** Result that shows whether the device is currently installing a feature update.
- **RemediationScanInProgress** Result that shows whether the device is currently scanning for updates.
- **RemediationTargetMachine** Result that shows whether this device is a candidate for remediation(s) that will fix update issues.
- **SetupMutexAvailable** Result that shows whether setup mutex is available or not.
- **SysPowerStatusAC** Result that shows whether system is on AC power or not.

#### **Microsoft.Windows.Remediation.Completed**

Enables tracking the completion of a process that remediates issues preventing security and quality updates.

The following fields are available:

- **CV** Correlation vector.
- **GlobalEventCounter** Client side counter which indicates ordering of events sent by the remediation system.
- **PackageVersion** Current package version of Remediation.
- **PluginName** Name of the specific remediation for each generic plugin event.
- **RemediationNoisyHammerTaskKickOffIsSuccess** Event that indicates the Update Assistant task has been started successfully.
- **Result** Indicates whether the remediation has completed.

#### **Microsoft.Windows.Remediation.RemediationShellMainExeEventId**

Enables tracking the ID of a process that remediates issues preventing security and quality updates.

The following fields are available:

- **CV** Correlation vector.
- **GlobalEventCounter** Client side counter which indicates ordering of events sent by the remediation system.
- **PackageVersion** Current package version of Remediation.
- **RemediationShellCanAcquireSedimentMutex** True if the remediation was able to acquire the sediment mutex. False if it is already running.
- **RemediationShellExecuteShellResult** Indicates if the remediation system completed without errors.
- **RemediationShellFoundDriverDII** Indicates whether the remediation system found its component files to run properly.
- **RemediationShellLoadedShellDriver** Indicates whether the remediation system loaded its component files to run properly.
- **RemediationShellLoadedShellFunction** Indicates whether the remediation system loaded the functions from its component files to run properly.

#### **Microsoft.Windows.Remediation.Started**

Enables tracking the start of a process that remediates issues preventing security and quality updates.

The following fields are available:

- **CV** Correlation vector.
- **GlobalEventCounter** Client side counter which indicates ordering of events sent by the remediation system.
- **PackageVersion** Current package version of Remediation.
- **PluginName** Name of the specific remediation for each generic plugin event.
- **Result** Results of the detection or perform action phases of the remediation system.

## Sediment Service events

### NOTE

Events from this provider are sent with the installation of KB4023057 and any subsequent Windows update. For details, see [this support article](#).

### Microsoft.Windows.SedimentService.Applicable

Indicates whether a given plugin is applicable.

The following fields are available:

- **CV** Correlation vector.
- **DetectedCondition** Boolean true if detect condition is true and perform action will be run.
- **GlobalEventCounter** Client side counter which indicates ordering of events.
- **IsSelfUpdateEnabledInOneSettings** True/False based on whether self update is enabled.
- **IsSelfUpdateNeeded** True/False based on whether a newer version is available.
- **PackageVersion** Version of the package.
- **PluginName** Name of the plugin specified for each generic plugin event.
- **Result** This is the HRESULT for detection or perform action phases of the plugin.

### Microsoft.Windows.SedimentService.Completed

Indicates whether a given plugin has completed its work.

The following fields are available:

- **CV** Correlation vector.
- **FailedReasons** String reason for any plugin failures.
- **GlobalEventCounter** Client side counter which indicates ordering of events.
- **PackageVersion** Current package version of Remediation.
- **PluginName** Name of the plugin specified for each generic plugin event.
- **Result** Result of the service execution.
- **SedimentServiceCheckTaskFunctional** Result of checking if the scheduled task is functional.
- **SedimentServiceCurrentBytes** Current number of bytes the service is consuming.
- **SedimentServiceKillService** True/False based on whether the service should be stopped.
- **SedimentServiceMaximumBytes** Maximum bytes the service can consume.
- **SedimentServiceRetrievedKillService** True/False whether the kill service information was retrieved.
- **SedimentServiceStopping** True/False indicating whether the service was found to be stopping.
- **SedimentServiceTaskFunctional** True/False if scheduled task is functional. If task is not functional this indicates plugins will be run.
- **SedimentServiceTotalIterations** Number of iterations service will wait before running again.

### Microsoft.Windows.SedimentService.Error

Indicates whether an error condition occurs in the plugin.

The following fields are available:

- **Message** String message containing information from the service.
- **PackageVersion** Version of the package.
- **HResult** Return value from the plugin result.

#### **Microsoft.Windows.SedimentService.FallbackError**

Indicates whether an error occurs for a fallback in the plugin.

The following fields are available:

- **s0** Fallback error level.
- **wilResult** Result for Windows Installer Logging function.

#### **Microsoft.Windows.SedimentService.Information**

General information returned from the plugin.

The following fields are available:

- **HResult** Result of the plugin execution.
- **Message** Information collected from the plugin based on the purpose of the plugin.
- **PackageVersion** Version of the package.

#### **Microsoft.Windows.SedimentService.Started**

Indicates that a given plugin has started.

The following fields are available:

- **CV** Correlation vector
- **GlobalEventCounter** Client side counter which indicates ordering of events.
- **PackageVersion** Version of the package.
- **PluginName** Name of the plugin running.
- **Result** Return code from the plugin result.

#### **Microsoft.Windows.SedimentService.wilResult**

Result from the windows internal library.

The following fields are available:

- **callContext** List of telemetry activities containing this error.
- **currentContextId** Identifier for the newest telemetry activity containing this error.
- **currentContextMessage** Custom message associated with the newest telemetry activity containing this error (if any).
- **currentContextName** Name of the newest telemetry activity containing this error.
- **failureType** Indicates what type of failure was observed (exception, returned error, logged error or fail fast).
- **failureId** Identifier assigned to this failure.
- **filename** The name of the source file where the error occurred.
- **hresult** Failure error code.
- **lineNumber** Line number within the source file where the error occurred.
- **message** Custom message associated with the failure (if any).
- **module** Name of the binary where the error occurred.
- **originatingContextId** Identifier for the oldest telemetry activity containing this error.
- **originatingContextMessage** Custom message associated with the oldest telemetry activity containing this

error (if any).

- **originatingContextName** Name of the oldest telemetry activity containing this error.
- **threadId** Identifier of the thread the error occurred on.

## Sediment Launcher events

### NOTE

Events from this provider are sent with the installation of KB4023057 and any subsequent Windows update. For details, see [this support article](#).

### Microsoft.Windows.SedimentLauncher.Applicable

Indicates whether a given plugin is applicable.

The following fields are available:

- **CV** Correlation vector.
- **DetectedCondition** Boolean true if detect condition is true and action will be run.
- **GlobalEventCounter** Client side counter which indicates ordering of events.
- **IsSelfUpdateEnabledInOneSettings** True/False based on whether self update is enabled.
- **IsSelfUpdateNeeded** True/False based on whether a newer version is available.
- **PackageVersion** Version of the package.
- **PluginName** Name of the plugin specified for each generic plugin event.
- **Result** This is the HRESULT for detection or perform action phases of the plugin.

### Microsoft.Windows.SedimentLauncher.Completed

Indicates whether a given plugin has completed its work.

The following fields are available:

- **CV** Correlation vector.
- **FailedReasons** String reason for any plugin failures.
- **GlobalEventCounter** Client side counter which indicates ordering of events.
- **PackageVersion** Current package version of Remediation.
- **PluginName** Name of the plugin specified for each generic plugin event.
- **Result** Result of the service execution.
- **SedLauncherExecutionResult** Final result of launcher running the plugins from the dll.

### Microsoft.Windows.SedimentLauncher.Error

Error occurred during execution of the plugin.

The following fields are available:

- **Message** Information message returned from a plugin containing only information internal to plugin execution.
- **PackageVersion** Version of the package.
- **HResult** Return value from the plugin result.

### Microsoft.Windows.SedimentLauncher.FallbackError

Error occurred during execution of the plugin fallback.

The following fields are available:

- **s0** Fallback error level for plugin.

- **wilResult** Result from executing Windows Installer Logging based function.

#### **Microsoft.Windows.SedimentLauncher.Information**

General information returned from the plugin.

The following fields are available:

- **HResult** Result of the plugin execution.
- **Message** Information collected from the plugin based on the purpose of the plugin.
- **PackageVersion** Version of the package.

#### **Microsoft.Windows.SedimentLauncher.Started**

Indicates that a given plugin has started.

The following fields are available:

- **CV** Correlation vector.
- **GlobalEventCounter** Client side counter which indicates ordering of events.
- **PackageVersion** Version of the package.
- **PluginName** Name of the plugin running.
- **Result** Return code from the plugin result.

#### **Microsoft.Windows.SedimentLauncher.wilResult**

Result from the windows internal library.

The following fields are available:

- **callContext** List of telemetry activities containing this error.
- **currentContextId** Identifier for the newest telemetry activity containing this error.
- **currentContextMessage** Custom message associated with the newest telemetry activity containing this error (if any).
- **currentContextName** Name of the newest telemetry activity containing this error.
- **failurecount** Number of failures seen.
- **failureType** Indicates what type of failure was observed (exception, returned error, logged error or fail fast).
- **failureId** Identifier assigned to this failure.
- **filename** The name of the source file where the error occurred.
- **function** Name of the function where the error occurred.
- **hresult** Failure error code.
- **lineNumber** Line number within the source file where the error occurred.
- **message** Custom message associated with the failure (if any).
- **module** Name of the binary where the error occurred.
- **originatingContextId** Identifier for the oldest telemetry activity containing this error.
- **originatingContextMessage** Custom message associated with the oldest telemetry activity containing this error (if any).
- **originatingContextName** Name of the oldest telemetry activity containing this error.
- **threadId** Identifier of the thread the error occurred on.

## Setup events

#### **SetupPlatformTel.SetupPlatformTelActivityStarted**

This event sends basic metadata about the update installation process generated by SetupPlatform to help keep Windows up to date.



The following fields are available:

- **Name** The name of the dynamic update type. Example: GDR driver

#### **SetupPlatformTel.SetupPlatformTelActivityEvent**

This event sends basic metadata about the SetupPlatform update installation process, to help keep Windows up-to-date

The following fields are available:

- **ActivityId** Provides a unique Id to correlate events that occur between a activity start event, and a stop event
- **ActivityName** Provides a friendly name of the package type that belongs to the ActivityId (Setup, LanguagePack, GDR, Driver, etc.)
- **FieldName** Retrieves the event name/data point. Examples: InstallStartTime, InstallEndtime, OverallResult etc.
- **GroupName** Retrieves the groupname the event belongs to. Example: Install Information, DU Information, Disk Space Information etc.
- **value** Value associated with the corresponding event name. For example, time-related events will include the system time

#### **SetupPlatformTel.SetupPlatformTelEvent**

This service retrieves events generated by SetupPlatform, the engine that drives the various deployment scenarios.

The following fields are available:

- **FieldName** Retrieves the event name/data point. Examples: InstallStartTime, InstallEndtime, OverallResult etc.
- **Value** Retrieves the value associated with the corresponding event name (Field Name). For example: For time related events this will include the system time.
- **GroupName** Retrieves the groupname the event belongs to. Example: Install Information, DU Information, Disk Space Information etc.

## Shared PC events

#### **Microsoft.Windows.SharedPC.AccountManager.DeleteUserAccount**

Activity for deletion of a user account for devices set up for Shared PC mode as part of the Transient Account Manager to help keep Windows up to date. Deleting unused user accounts on shared devices frees up disk space to improve Windows Update success rates.

The following fields are available:

- **accountType** The type of account that was deleted. Example: AD, AAD, or Local
- **userSid** The security identifier of the account.
- **wilActivity** Windows Error Reporting data collected when there is a failure in deleting a user account with the Transient Account Manager.

#### **Microsoft.Windows.SharedPC.AccountManager.SinglePolicyEvaluation**

Activity for run of the Transient Account Manager that determines if any user accounts should be deleted for devices set up for Shared PC mode to help keep Windows up to date. Deleting unused user accounts on shared devices frees up disk space to improve Windows Update success rates

The following fields are available:

- **wilActivity** Windows Error Reporting data collected when there is a failure in evaluating accounts to be deleted with the Transient Account Manager.
- **totalAccountCount** The number of accounts on a device after running the Transient Account Manager policies.

- **evaluationTrigger** When was the Transient Account Manager policies ran? Example: At log off or during maintenance hours

## Software update events

### SoftwareUpdateClientTelemetry.UpdateDetected

This event sends data about an AppX app that has been updated from the Microsoft Store, including what app needs an update and what version/architecture is required, in order to understand and address problems with apps getting required updates.

The following fields are available:

- **ApplicableUpdateInfo** Metadata for the updates which were detected as applicable
- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client
- **NumberOfApplicableUpdates** The number of updates which were ultimately deemed applicable to the system after the detection process is complete
- **RelatedCV** The previous Correlation Vector that was used before swapping with a new one
- **WUDeviceID** The unique device ID controlled by the software distribution client
- **IntentPFNs** Intended application-set metadata for atomic update scenarios.
- **ServiceGuid** An ID which represents which service the software distribution client is connecting to (Windows Update, Microsoft Store, etc.)

### SoftwareUpdateClientTelemetry.SLSDiscovery

This event sends data about the ability of Windows to discover the location of a backend server with which it must connect to perform updates or content acquisition, in order to determine disruptions in availability of update services and provide context for Windows Update errors.

The following fields are available:

- **EventScenario** Indicates the purpose of sending this event - whether because the software distribution just started checking for content, or whether it was cancelled, succeeded, or failed
- **HResult** Indicates the result code of the event (success, cancellation, failure code HRESULT)
- **IsBackground** Indicates whether the SLS discovery event took place in the foreground or background
- **NextExpirationTime** Indicates when the SLS cab expires
- **ServiceID** An ID which represents which service the software distribution client is connecting to (Windows Update, Microsoft Store, etc.)
- **SusClientID** The unique device ID controlled by the software distribution client
- **UrlPath** Path to the SLS cab that was downloaded
- **WUAVersion** The version number of the software distribution client

### SoftwareUpdateClientTelemetry.Commit

This event sends data on whether the Update Service has been called to execute an upgrade, to help keep Windows up to date.

The following fields are available:

- **BiosFamily** The family of the BIOS (Basic Input Output System).
- **BiosName** The name of the device BIOS.
- **BiosReleaseDate** The release date of the device BIOS.
- **BiosSKUNumber** The sku number of the device BIOS.
- **BIOSVendor** The vendor of the BIOS.
- **BiosVersion** The version of the BIOS.

- **BundleId** Identifier associated with the specific content bundle; should not be all zeros if the bundleID was found.
- **ClientVersion** The version number of the software distribution client.
- **DeviceModel** What is the device model.
- **EventInstanceId** A globally unique identifier for event instance.
- **EventScenario** State of call
- **EventType** "Possible values are ""Child"", ""Bundle"", or ""Driver""."
- **HandlerType** Indicates the kind of content (app, driver, windows patch, etc.)
- **RevisionNumber** Unique revision number of Update
- **ServerId** Identifier for the service to which the software distribution client is connecting, such as Windows Update and Microsoft Store.
- **SystemBIOSMajorRelease** Major version of the BIOS.
- **SystemBIOSMinorRelease** Minor version of the BIOS.
- **UpdateId** Unique Update ID
- **WUDeviceID** UniqueDeviceID
- **BundleRevisionNumber** Identifies the revision number of the content bundle
- **FlightId** The specific id of the flight the device is getting
- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client

#### SoftwareUpdateClientTelemetry.DownloadCheckpoint

This event provides a checkpoint between each of the Windows Update download phases for UUP content

The following fields are available:

- **EventScenario** Indicates the purpose of sending this event - whether because the software distribution just started checking for content, or whether it was cancelled, succeeded, or failed
- **ExtendedStatusCode** Secondary error code for certain scenarios where StatusCode wasn't specific enough
- **FileId** A hash that uniquely identifies a file
- **FileName** Name of the downloaded file
- **RelatedCV** The previous Correlation Vector that was used before swapping with a new one
- **StatusCode** Indicates the result of a CheckForUpdates event (success, cancellation, failure code HRESULT)
- **EventType** "Possible values are ""Child"", ""Bundle"", ""Release"" or ""Driver"""
- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client
- **ClientVersion** The version number of the software distribution client
- **FlightId** The unique identifier for each flight
- **RevisionNumber** Unique revision number of Update
- **ServiceGuid** An ID which represents which service the software distribution client is checking for content (Windows Update, Microsoft Store, etc.)
- **UpdateId** Unique Update ID
- **WUDeviceID** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue

#### SoftwareUpdateClientTelemetry.UpdateMetadataIntegrity

This event identifies whether updates have been tampered with and protects against man-in-the-middle attacks.

The following fields are available:

- **EventScenario** The purpose of this event, such as scan started, scan succeeded, or scan failed.
- **ExtendedStatusCode** The secondary status code of the event.

- **LeafCertId** Integral ID from the FragmentSigning data for certificate that failed.
- **MetadataIntegrityMode** The mode of the transport metadata integrity check. 0 = unknown; 1 = ignore; 2 = audit; 3 = enforce
- **MetadataSignature** A base64-encoded string of the signature associated with the update metadata (specified by revision ID).
- **RevisionId** The revision ID for a specific piece of content.
- **RevisionNumber** The revision number for a specific piece of content.
- **ServiceGuid** Identifies the service to which the software distribution client is connected, Example: Windows Update or Microsoft Store
- **SHA256OfLeafCertPublicKey** A base64 encoding of the hash of the Base64CertData in the FragmentSigning data of the leaf certificate.
- **SHA256OfTimestampToken** A base64-encoded string of hash of the timestamp token blob.
- **SignatureAlgorithm** The hash algorithm for the metadata signature.
- **StatusCode** The status code of the event.
- **TimestampTokenId** The time this was created. It is encoded in a timestamp blob and will be zero if the token is malformed.
- **UpdateId** The update ID for a specific piece of content.
- **TimestampTokenCertThumbprint** "The thumbprint of the encoded timestamp token. "
- **ValidityWindowInDays** The validity window that's in effect when verifying the timestamp.
- **ListOfSHA256OfIntermediateCerData** A semicolon delimited list of base64 encoding of hashes for the Base64CerData in the FragmentSigning data of an intermediate certificate.
- **RawMode** The raw unparsed mode string from the SLS response. This field is null if not applicable.
- **RawValidityWindowInDays** The raw unparsed validity window string in days of the timestamp token. This field is null if not applicable.
- **SHA256OfLeafCerData** A base64 encoding of the hash for the Base64CerData in the FragmentSigning data of the leaf certificate.
- **EndpointUrl** The endpoint URL where the device obtains update metadata. This is used to distinguish between test, staging, and production environments.
- **SLSPrograms** A test program to which a device may have opted in. Example: Insider Fast

### SoftwareUpdateClientTelemetry.Download

This event sends tracking data about the software distribution client download of the content for that update, to help keep Windows up to date.

The following fields are available:

- **ActiveDownloadTime** How long the download took, in seconds, excluding time where the update wasn't actively being downloaded.
- **AppXBlockHashValidationFailureCount** A count of the number of blocks that have failed validation after being downloaded.
- **AppXDownloadScope** Indicates the scope of the download for application content. For streaming install scenarios, AllContent - non-streaming download, RequiredOnly - streaming download requested content required for launch, AutomaticOnly - streaming download requested automatic streams for the app, and Unknown - for events sent before download scope is determined by the Windows Update client.
- **BiosFamily** The family of the BIOS (Basic Input Output System).
- **BiosName** The name of the device BIOS.
- **BiosReleaseDate** The release date of the device BIOS.
- **BiosSKUNumber** The sku number of the device BIOS.
- **BIOSVendor** The vendor of the BIOS.
- **BiosVersion** The version of the BIOS.

- **BundleBytesDownloaded** How many bytes were downloaded for the specific content bundle.
- **BundleId** Identifier associated with the specific content bundle; should not be all zeros if the bundleID was found.
- **BundleRepeatFailFlag** Indicates whether this particular update bundle had previously failed to download.
- **BundleRevisionNumber** Identifies the revision number of the content bundle.
- **BytesDownloaded** How many bytes were downloaded for an individual piece of content (not the entire bundle).
- **CachedEngineVersion** For self-initiated healing, the version of the SIH engine that is cached on the device. If the SIH engine does not exist, the value is null.
- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client.
- **CbsDownloadMethod** Indicates whether the download was a full-file download or a partial/delta download.
- **CDNCountryCode** Two letter country abbreviation for the CDN's location.
- **CDNId** ID which defines which CDN the software distribution client downloaded the content from.
- **ClientManagedByWSUSServer** Indicates whether the client is managed by Windows Server Update Services (WSUS).
- **ClientVersion** The version number of the software distribution client.
- **CurrentMobileOperator** The mobile operator the device is currently connected to.
- **DeviceModel** What is the device model.
- **DeviceOEM** What OEM does this device belong to.
- **DownloadPriority** Indicates whether a download happened at background, normal, or foreground priority.
- **DownloadScenarioId** A unique ID for a given download used to tie together WU and DO events.
- **DownloadType** Differentiates the download type of SIH downloads between Metadata and Payload downloads.
- **Edition** Indicates the edition of Windows being used.
- **EventInstanceId** A globally unique identifier for event instance.
- **EventNamespaceID** Indicates whether the event succeeded or failed. Has the format EventType+Event where Event is Succeeded, Cancelled, Failed, etc.
- **EventScenario** Indicates the purpose of sending this event - whether because the software distribution just started downloading content, or whether it was cancelled, succeeded, or failed.
- **EventType** Possible values are Child, Bundle, or Driver.
- **ExtendedStatusCode** Secondary error code for certain scenarios where StatusCode wasn't specific enough.
- **FeatureUpdatePause** Indicates whether feature OS updates are paused on the device.
- **FlightBranch** The branch that a device is on if participating in flighting (pre-release builds).
- **FlightBuildNumber** If this download was for a flight (pre-release build), this indicates the build number of that flight.
- **FlightId** The specific id of the flight (pre-release build) the device is getting.
- **FlightRing** The ring (speed of getting builds) that a device is on if participating in flighting (pre-release builds).
- **HandlerType** Indicates what kind of content is being downloaded (app, driver, windows patch, etc.).
- **HardwareId** If this download was for a driver targeted to a particular device model, this ID indicates the model of the device.
- **HomeMobileOperator** The mobile operator that the device was originally intended to work with.
- **HostName** The hostname URL the content is downloading from.
- **IPVersion** Indicates whether the download took place over IPv4 or IPv6.
- **IsAOACDevice** Is it Always On, Always Connected?
- **IsDependentSet** Indicates whether a driver is a part of a larger System Hardware/Firmware Update
- **IsWUfBDualScanEnabled** Indicates if Windows Update for Business dual scan is enabled on the device.

- **IsWUfBEnabled** Indicates if Windows Update for Business is enabled on the device.
- **NetworkCostBitMask** Indicates what kind of network the device is connected to (roaming, metered, over data cap, etc.)
- **NetworkRestrictionStatus** "More general version of NetworkCostBitMask, specifying whether Windows considered the current network to be ""metered.""
- **PackageFullName** The package name of the content.
- **PhonePreviewEnabled** Indicates whether a phone was opted-in to getting preview builds, prior to flighting (pre-release builds) being introduced.
- **PlatformRole** The PowerPlatformRole as defined on MSDN
- **ProcessName** The process name of the caller who initiated API calls, in the event where CallerApplicationName was not provided.
- **ProcessorArchitecture** Processor architecture of the system (x86, AMD64, ARM).
- **QualityUpdatePause** Indicates whether quality OS updates are paused on the device.
- **RelatedCV** The previous Correlation Vector that was used before swapping with a new one
- **RepeatFailFlag** Indicates whether this specific piece of content had previously failed to download.
- **RevisionNumber** Identifies the revision number of this specific piece of content.
- **ServiceGuid** An ID which represents which service the software distribution client is installing content for (Windows Update, Microsoft Store, etc.).
- **Setup360Phase** If the download is for an operating system upgrade, this datapoint indicates which phase of the upgrade is underway.
- **ShippingMobileOperator** The mobile operator that a device shipped on.
- **StatusCode** Indicates the result of a Download event (success, cancellation, failure code HRESULT).
- **SystemBIOSMajorRelease** Major version of the BIOS.
- **SystemBIOSMinorRelease** Minor version of the BIOS.
- **TargetGroupID** For drivers targeted to a specific device model, this ID indicates the distribution group of devices receiving that driver.
- **TargetingVersion** For drivers targeted to a specific device model, this is the version number of the drivers being distributed to the device.
- **TargetMetadataVersion** For self-initiated healing, this is the target version of the SIH engine to download (if needed). If not, the value is null.
- **ThrottlingServiceHRESULT** Result code (success/failure) while contacting a web service to determine whether this device should download content yet.
- **TimeToEstablishConnection** Time (in ms) it took to establish the connection prior to beginning downloaded.
- **TotalExpectedBytes** The total count of bytes that the download is expected to be.
- **UpdateId** An identifier associated with the specific piece of content.
- **UpdateImportance** Indicates whether a piece of content was marked as Important, Recommended, or Optional.
- **UsedDO** Whether the download used the delivery optimization service.
- **UsedSystemVolume** Indicates whether the content was downloaded to the device's main system storage drive, or an alternate storage drive.
- **WUDeviceID** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue.
- **WUSetting** Indicates the users' current updating settings.

#### **SoftwareUpdateClientTelemetry.CheckForUpdates**

This event sends tracking data about the software distribution client check for content that is applicable to a device, to help keep Windows up to date

The following fields are available:

- **ActivityMatchingId** Contains a unique ID identifying a single CheckForUpdates session from initialization to completion.
- **AllowCachedResults** Indicates if the scan allowed using cached results.
- **BiosFamily** The family of the BIOS (Basic Input Output System).
- **BiosName** The name of the device BIOS.
- **BiosReleaseDate** The release date of the device BIOS.
- **BiosSKUNumber** The sku number of the device BIOS.
- **BIOSVendor** The vendor of the BIOS.
- **BiosVersion** The version of the BIOS.
- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client.
- **CapabilityDetectoidGuid** The GUID for a hardware applicability detectoid that could not be evaluated.
- **CDNCountryCode** Two letter country abbreviation for the CDN's location.
- **CDNId** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue.
- **ClientVersion** The version number of the software distribution client.
- **CurrentMobileOperator** The mobile operator the device is currently connected to.
- **DeviceModel** What is the device model.
- **DriverError** The error code hit during a driver scan. This is 0 if no error was encountered.
- **EventInstanceId** A globally unique identifier for event instance.
- **EventScenario** Indicates the purpose of sending this event - whether because the software distribution just started checking for content, or whether it was cancelled, succeeded, or failed.
- **ExtendedMetadataCabUrl** Hostname that is used to download an update.
- **ExtendedStatusCode** Secondary error code for certain scenarios where StatusCode wasn't specific enough.
- **FailedUpdateGuids** The GUIDs for the updates that failed to be evaluated during the scan.
- **FailedUpdatesCount** The number of updates that failed to be evaluated during the scan.
- **FlightBranch** The branch that a device is on if participating in flighting (pre-release builds).
- **FlightRing** The ring (speed of getting builds) that a device is on if participating in flighting (pre-release builds).
- **HomeMobileOperator** The mobile operator that the device was originally intended to work with.
- **IPVersion** Indicates whether the download took place over IPv4 or IPv6
- **IsWUfBDualScanEnabled** Indicates if Windows Update for Business dual scan is enabled on the device.
- **IsWUfBEnabled** Indicates if Windows Update for Business is enabled on the device.
- **MetadataIntegrityMode** The mode of the update transport metadata integrity check. 0-Unknown, 1-Ignoe, 2-Audit, 3-Enforce
- **MSIError** The last error that was encountered during a scan for updates.
- **NetworkConnectivityDetected** Indicates the type of network connectivity that was detected. 0 - IPv4, 1 - IPv6
- **NumberOfApplicationsCategoryScanEvaluated** The number of categories (apps) for which an app update scan checked
- **NumberOfLoop** The number of round trips the scan required
- **NumberOfNewUpdatesFromServiceSync** The number of updates which were seen for the first time in this scan
- **NumberOfUpdatesEvaluated** The total number of updates which were evaluated as a part of the scan
- **NumFailedMetadataSignatures** The number of metadata signatures checks which failed for new metadata synced down.
- **Online** Indicates if this was an online scan.
- **PhonePreviewEnabled** Indicates whether a phone was getting preview build, prior to flighting (pre-release

builds) being introduced.

- **ProcessName** The process name of the caller who initiated API calls, in the event where CallerApplicationName was not provided.
- **RelatedCV** The previous Correlation Vector that was used before swapping with a new one
- **ScanDurationInSeconds** The number of seconds a scan took
- **ScanEnqueueTime** The number of seconds it took to initialize a scan
- **ServiceGuid** An ID which represents which service the software distribution client is checking for content (Windows Update, Microsoft Store, etc.).
- **ServiceUrl** The environment URL a device is configured to scan with
- **ShippingMobileOperator** The mobile operator that a device shipped on.
- **StatusCode** Indicates the result of a CheckForUpdates event (success, cancellation, failure code HRESULT).
- **SyncType** Describes the type of scan the event was
- **SystemBIOSMajorRelease** Major version of the BIOS.
- **SystemBIOSMinorRelease** Minor version of the BIOS.
- **TotalNumMetadataSignatures** The total number of metadata signatures checks done for new metadata that was synced down.
- **WUDeviceID** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue.
- **ApplicableUpdateInfo** Metadata for the updates which were detected as applicable
- **NumberOfApplicableUpdates** The number of updates which were ultimately deemed applicable to the system after the detection process is complete
- **WebServiceRetryMethods** Web service method requests that needed to be retried to complete operation.
- **BranchReadinessLevel** The servicing branch configured on the device.
- **DeferralPolicySources** Sources for any update deferral policies defined (GPO = 0x10, MDM = 0x100, Flight = 0x1000, UX = 0x10000).
- **DeferredUpdates** Update IDs which are currently being deferred until a later time
- **DriverExclusionPolicy** Indicates if the policy for not including drivers with Windows Update is enabled.
- **FeatureUpdateDeferral** The deferral period configured for feature OS updates on the device (in days).
- **FeatureUpdatePause** Indicates whether feature OS updates are paused on the device.
- **FeatureUpdatePausePeriod** The pause duration configured for feature OS updates on the device (in days).
- **QualityUpdateDeferral** The deferral period configured for quality OS updates on the device (in days).
- **QualityUpdatePause** Indicates whether quality OS updates are paused on the device.
- **QualityUpdatePausePeriod** The pause duration configured for quality OS updates on the device (in days).
- **IntentPFNs** Intended application-set metadata for atomic update scenarios.
- **PausedUpdates** A list of UpdateIds which that currently being paused.
- **PauseFeatureUpdatesEndTime** If feature OS updates are paused on the device, this is the date and time for the end of the pause time window.
- **PauseFeatureUpdatesStartTime** If feature OS updates are paused on the device, this is the date and time for the beginning of the pause time window.
- **PauseQualityUpdatesEndTime** If quality OS updates are paused on the device, this is the date and time for the end of the pause time window.
- **PauseQualityUpdatesStartTime** If quality OS updates are paused on the device, this is the date and time for the beginning of the pause time window.
- **CachedEngineVersion** For self-initiated healing, the version of the SIH engine that is cached on the device. If the SIH engine does not exist, the value is null.
- **TargetMetadataVersion** For self-initiated healing, this is the target version of the SIH engine to download (if needed). If not, the value is null.
- **Context** Gives context on where the error has occurred. Example: AutoEnable, GetSLSData, AddService, Misc,



or Unknown

- **DriverSyncPassPerformed** Were drivers scanned this time?

### **SoftwareUpdateClientTelemetry.Install**

This event sends tracking data about the software distribution client installation of the content for that update, to help keep Windows up to date.

The following fields are available:

- **BiosFamily** The family of the BIOS (Basic Input Output System).
- **BiosName** The name of the device BIOS.
- **BiosReleaseDate** The release date of the device BIOS.
- **BiosSKUNumber** The sku number of the device BIOS.
- **BIOSVendor** The vendor of the BIOS.
- **BiosVersion** The version of the BIOS.
- **BundleBytesDownloaded** How many bytes were downloaded for the specific content bundle?
- **BundleId** Identifier associated with the specific content bundle; should not be all zeros if the bundleID was found.
- **BundleRepeatFailFlag** Has this particular update bundle previously failed to install?
- **BundleRevisionNumber** Identifies the revision number of the content bundle.
- **CachedEngineVersion** For self-initiated healing, the version of the SIH engine that is cached on the device. If the SIH engine does not exist, the value is null.
- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client.
- **CbsDownloadMethod** Was the download a full download or a partial download?
- **ClientManagedByWSUSServer** Is the client managed by Windows Server Update Services (WSUS)?
- **ClientVersion** The version number of the software distribution client.
- **CSLErrorType** The stage of CBS installation where it failed.
- **CurrentMobileOperator** Mobile operator that device is currently connected to.
- **DeviceModel** What is the device model.
- **DeviceOEM** What OEM does this device belong to.
- **DownloadPriority** The priority of the download activity.
- **DownloadScenarioId** A unique ID for a given download used to tie together WU and DO events.
- **DriverPingBack** Contains information about the previous driver and system state.
- **Edition** Indicates the edition of Windows being used.
- **EventInstanceId** A globally unique identifier for event instance.
- **EventNamespaceID** Indicates whether the event succeeded or failed. Has the format EventType+Event where Event is Succeeded, Cancelled, Failed, etc.
- **EventScenario** Indicates the purpose of sending this event - whether because the software distribution just started installing content, or whether it was cancelled, succeeded, or failed.
- **EventType** Possible values are Child, Bundle, or Driver.
- **ExtendedErrorCode** The extended error code.
- **ExtendedStatusCode** Secondary error code for certain scenarios where StatusCode wasn't specific enough.
- **FeatureUpdatePause** Are feature OS updates paused on the device?
- **FlightBranch** The branch that a device is on if participating in the Windows Insider Program.
- **FlightBuildNumber** If this installation was for a Windows Insider build, this is the build number of that build.
- **FlightId** The specific ID of the Windows Insider build the device is getting.
- **FlightRing** The ring that a device is on if participating in the Windows Insider Program.
- **HandlerType** Indicates what kind of content is being installed. Example: app, driver, Windows update

- **HardwareId** If this install was for a driver targeted to a particular device model, this ID indicates the model of the device.
- **HomeMobileOperator** The mobile operator that the device was originally intended to work with.
- **IntentPFNs** Intended application-set metadata for atomic update scenarios.
- **IsAOACDevice** Is it Always On, Always Connected? (Mobile device usage model)
- **IsDependentSet** Is the driver part of a larger System Hardware/Firmware update?
- **IsFinalOutcomeEvent** Does this event signal the end of the update/upgrade process?
- **IsFirmware** Is this update a firmware update?
- **IsSuccessFailurePostReboot** Did it succeed and then fail after a restart?
- **IsWUfBDualScanEnabled** Is Windows Update for Business dual scan enabled on the device?
- **IsWUfBEnabled** Is Windows Update for Business enabled on the device?
- **MergedUpdate** Was the OS update and a BSP update merged for installation?
- **MsiAction** The stage of MSI installation where it failed.
- **MsiProductCode** The unique identifier of the MSI installer.
- **PackageFullName** The package name of the content being installed.
- **PhonePreviewEnabled** Indicates whether a phone was getting preview build, prior to flighting being introduced.
- **PlatformRole** The PowerPlatformRole as defined on MSDN.
- **ProcessName** The process name of the caller who initiated API calls, in the event where CallerApplicationName was not provided.
- **ProcessorArchitecture** Processor architecture of the system (x86, AMD64, ARM).
- **QualityUpdatePause** Are quality OS updates paused on the device?
- **RelatedCV** The previous Correlation Vector that was used before swapping with a new one
- **RepeatFailFlag** Indicates whether this specific piece of content had previously failed to install.
- **RepeatSuccessInstallFlag** Indicates whether this specific piece of content had previously installed successfully, for example if another user had already installed it.
- **RevisionNumber** The revision number of this specific piece of content.
- **ServiceGuid** An ID which represents which service the software distribution client is installing content for (Windows Update, Microsoft Store, etc.).
- **Setup360Phase** If the install is for an operating system upgrade, indicates which phase of the upgrade is underway.
- **ShippingMobileOperator** The mobile operator that a device shipped on.
- **StatusCode** Indicates the result of an installation event (success, cancellation, failure code HRESULT).
- **SystemBIOSMajorRelease** Major version of the BIOS.
- **SystemBIOSMinorRelease** Minor version of the BIOS.
- **TargetGroupId** For drivers targeted to a specific device model, this ID indicates the distribution group of devices receiving that driver.
- **TargetingVersion** For drivers targeted to a specific device model, this is the version number of the drivers being distributed to the device.
- **TransactionCode** The ID which represents a given MSI installation
- **UpdateId** Unique update ID
- **UpdateImportance** Indicates whether a piece of content was marked as Important, Recommended, or Optional.
- **UsedSystemVolume** Indicates whether the content was downloaded and then installed from the device's main system storage drive, or an alternate storage drive.
- **WUDeviceID** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue.
- **WUSetting** Indicates the user's current updating settings.

## SoftwareUpdateClientTelemetry.DownloadHeartbeat

This event allows tracking of ongoing downloads and contains data to explain the current state of the download

The following fields are available:

- **BundleID** Identifier associated with the specific content bundle. If this value is found, it shouldn't report as all zeros
- **BytesTotal** Total bytes to transfer for this content
- **BytesTransferred** Total bytes transferred for this content at the time of heartbeat
- **ConnectionStatus** Indicates the connectivity state of the device at the time of heartbeat
- **CurrentError** Last (transient) error encountered by the active download
- **DownloadFlags** Flags indicating if power state is ignored
- **DownloadState** Current state of the active download for this content (queued, suspended, or progressing)
- **IsNetworkMetered** "Indicates whether Windows considered the current network to be ?metered""
- **MOAppDownloadLimit** Mobile operator cap on size of application downloads, if any
- **MOUpdateDownloadLimit** Mobile operator cap on size of operating system update downloads, if any
- **PowerState** Indicates the power state of the device at the time of heartbeat (DC, AC, Battery Saver, or Connected Standby)
- **RelatedCV** "The previous correlation vector that was used by the client, before swapping with a new one "
- **ResumeCount** Number of times this active download has resumed from a suspended state
- **ServiceID** "Identifier for the service to which the software distribution client is connecting (Windows Update, Microsoft Store, etc) "
- **SuspendCount** Number of times this active download has entered a suspended state
- **SuspendReason** Last reason for why this active download entered a suspended state
- **CallerApplicationName** Name provided by the caller who initiated API calls into the software distribution client
- **ClientVersion** The version number of the software distribution client
- **EventType** "Possible values are ""Child"", ""Bundle"", or ""Driver"""
- **FlightId** The unique identifier for each flight
- **RevisionNumber** Identifies the revision number of this specific piece of content
- **ServiceGuid** Identifier for the service to which the software distribution client is connecting (Windows Update, Microsoft Store, etc)
- **UpdateId** "Identifier associated with the specific piece of content "
- **WUDeviceID** "Unique device id controlled by the software distribution client "

## Update events

### Update360Telemetry.UpdateAgentPostRebootResult

This event collects information for both Mobile and Desktop regarding the post reboot phase of the new UUP (Unified Update Platform) update scenario

The following fields are available:

- **ErrorCode** The error code returned for the current post reboot phase
- **FlightId** The unique identifier for each flight
- **ObjectId** Unique value for each Update Agent mode
- **RelatedCV** Correlation vector value generated from the latest USO scan
- **Result** Indicates the HRESULT
- **ScenarioId** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate

- **SessionId** Unique value for each Update Agent mode attempt
- **UpdateId** Unique ID for each update
- **PostRebootResult** Indicates the Hresult

#### Update360Telemetry.UpdateAgent\_Initialize

This event sends data during the initialize phase of updating Windows.

The following fields are available:

- **ErrorCode** The error code returned for the current initialize phase.
- **FlightId** Unique ID for each flight.
- **FlightMetadata** Contains the FlightId and the build being flighted.
- **ObjectId** Unique value for each Update Agent mode.
- **RelatedCV** Correlation vector value generated from the latest USO scan.
- **ScenarioId** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **SessionData** Contains instructions to update agent for processing FODs and DUICs (Null for other scenarios).
- **SessionId** Unique value for each Update Agent mode attempt .
- **UpdateId** Unique ID for each update.
- **Result** Result of the initialize phase of update. 0 = Succeeded, 1 = Failed, 2 = Cancelled, 3 = Blocked, 4 = BlockCancelled

#### Update360Telemetry.UpdateAgent\_DownloadRequest

This event sends data during the download request phase of updating Windows.

The following fields are available:

- **ErrorCode** The error code returned for the current download request phase.
- **ObjectId** Unique value for each Update Agent mode.
- **PackageCountOptional** Number of optional packages requested.
- **PackageCountRequired** Number of required packages requested.
- **PackageCountTotal** Total number of packages needed.
- **RelatedCV** Correlation vector value generated from the latest USO scan.
- **ScenarioId** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **SessionId** Unique value for each Update Agent mode attempt.
- **PackageSizeCanonical** Size of canonical packages in bytes
- **PackageSizeDiff** Size of diff packages in bytes
- **PackageSizeExpress** Size of express packages in bytes
- **Result** Result of the download request phase of update.
- **FlightId** Unique ID for each flight.
- **UpdateId** Unique ID for each update.
- **PackageCountTotalCanonical** Total number of canonical packages.
- **PackageCountTotalDiff** Total number of diff packages.
- **PackageCountTotalExpress** Total number of express packages.
- **DeletedCorruptFiles** Indicates if UpdateAgent found any corrupt payload files and whether the payload was deleted.
- **RangeRequestState** Represents the state of the download range request.

#### Update360Telemetry.UpdateAgent\_Install

This event sends data during the install phase of updating Windows.

The following fields are available:

- **ErrorCode** The error code returned for the current install phase.
- **ObjectId** Unique value for each Update Agent mode.
- **RelatedCV** Correlation vector value generated from the latest scan.
- **ScenarioId** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **SessionId** Unique value for each Update Agent mode attempt.
- **Result** "Result of the install phase of update. 0 = Succeeded 1 = Failed, 2 = Cancelled, 3 = Blocked, 4 = BlockCancelled "
- **FlightId** Unique ID for each flight.
- **UpdateId** Unique ID for each update.

#### **Update360Telemetry.UpdateAgent\_ModeStart**

This event sends data for the start of each mode during the process of updating Windows.

The following fields are available:

- **Mode** Indicates that the Update Agent mode that has started. 1 = Initialize, 2 = DownloadRequest, 3 = Install, 4 = Commit
- **ObjectId** Unique value for each Update Agent mode.
- **RelatedCV** The correlation vector value generated from the latest scan.
- **ScenarioId** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **SessionId** Unique value for each Update Agent mode attempt.
- **FlightId** Unique ID for each flight.
- **UpdateId** Unique ID for each update.

#### **Update360Telemetry.UpdateAgent\_SetupBoxLaunch**

This event sends data during the launching of the setup box when updating Windows.

The following fields are available:

- **ObjectId** Unique value for each Update Agent mode.
- **Quiet** Indicates whether setup is running in quiet mode. 0 = false 1 = true
- **RelatedCV** Correlation vector value generated from the latest scan.
- **ScenarioId** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **SessionId** Unique value for each Update Agent mode attempt.
- **FlightId** Unique ID for each flight.
- **UpdateId** Unique ID for each update.
- **SetupMode** Setup mode 1 = predownload, 2 = install, 3 = finalize
- **SandboxSize** The size of the sandbox folder on the device.

## Update notification events

#### **Microsoft.Windows.UpdateNotificationPipeline.JavascriptJavascriptCriticalGenericMessage**

This event indicates that Javascript is reporting a schema and a set of values for critical diagnostic data.

The following fields are available:

- **CampaignConfigVersion** Configuration version for the current campaign
- **CampaignID** Currently campaign that's running on UNP

- **ConfigCatalogVersion** Current catalog version of UNP
- **ContentVersion** Content version for the current campaign on UNP
- **CV** Correlation vector
- **DetectorVersion** Most recently run detector version for the current campaign on UNP
- **GlobalEventCounter** Client-side counter that indicates the event ordering sent by the user
- **key1** Interaction data for the UI
- **key10** Interaction data for the UI
- **key11** Interaction data for the UI
- **key12** Interaction data for the UI
- **key13** Interaction data for the UI
- **key14** Interaction data for the UI
- **key15** Interaction data for the UI
- **key16** Interaction data for the UI
- **key17** Interaction data for the UI
- **key18** Interaction data for the UI
- **key19** Interaction data for the UI
- **key2** Interaction data for the UI
- **key20** Interaction data for the UI
- **key21** Interaction data for the UI
- **key22** Interaction data for the UI
- **key23** Interaction data for the UI
- **key24** Interaction data for the UI
- **key25** Interaction data for the UI
- **key26** Interaction data for the UI
- **key27** Interaction data for the UI
- **key28** Interaction data for the UI
- **key29** Interaction data for the UI
- **key3** Interaction data for the UI
- **key30** Interaction data for the UI
- **key4** Interaction data for the UI
- **key5** Interaction data for the UI
- **key6** Interaction data for the UI
- **key7** Interaction data for the UI
- **key8** Interaction data for the UI
- **key9** Interaction data for the UI
- **PackageVersion** Current package version of UNP
- **schema** Type of UI interaction

#### **Microsoft.Windows.UpdateNotificationPipeline.UNPCampaignHeartbeat**

This event is sent at the start of each campaign, to be used as a heartbeat

The following fields are available:

- **CampaignConfigVersion** Configuration version for the current campaign
- **CampaignID** Currently campaign that's running on UNP
- **ConfigCatalogVersion** Current catalog version of UNP
- **ContentVersion** Content version for the current campaign on UNP
- **CV** Correlation vector

- **DetectorVersion** Most recently run detector version for the current campaign on UNP
- **GlobalEventCounter** Client-side counter that indicates the event ordering sent by the user
- **PackageVersion** Current UNP package version

#### **Microsoft.Windows.UpdateNotificationPipeline.UNPCampaignManagerCleaningCampaign**

This event indicates that the Campaign Manager is cleaning up the campaign content

The following fields are available:

- **CampaignConfigVersion** Configuration version for the current campaign
- **CampaignID** Current campaign that's running on UNP
- **ConfigCatalogVersion** Current catalog version of UNP
- **ContentVersion** Content version for the current campaign on UNP
- **CV** Correlation vector
- **DetectorVersion** Most recently run detector version for the current campaign on UNP
- **GlobalEventCounter** Client-side counter that indicates the event ordering sent by the user
- **PackageVersion** Current UNP package version

#### **Microsoft.Windows.UpdateNotificationPipeline.UnpCampaignManagerGetIsCampaignCompleteFailed**

This event is sent when a campaign completion status query fails

The following fields are available:

- **CampaignConfigVersion** Configuration version for the current campaign
- **CampaignID** Current campaign that's running on UNP
- **ConfigCatalogVersion** Current catalog version of UNP
- **ContentVersion** Content version for the current campaign on UNP
- **CV** Correlation vector
- **DetectorVersion** Most recently run detector version for the current campaign on UNP
- **GlobalEventCounter** Client-side counter that indicates the event ordering sent by the user
- **hresult** HRESULT of the failure
- **PackageVersion** Current UNP package version

#### **Microsoft.Windows.UpdateNotificationPipeline.UNPCampaignManagerHeartbeat**

This event is sent at the start of the CampaignManager event and is intended to be used as a heartbeat

The following fields are available:

- **CampaignConfigVersion** Configuration version for the current campaign
- **CampaignID** Currently campaign that's running on UNP
- **ConfigCatalogVersion** Current catalog version of UNP
- **ContentVersion** Content version for the current campaign on UNP
- **CV** Correlation vector
- **DetectorVersion** Most recently run detector version for the current campaign on UNP
- **GlobalEventCounter** Client-side counter that indicates the event ordering sent by the user
- **PackageVersion** Current UNP package version

#### **Microsoft.Windows.UpdateNotificationPipeline.UnpCampaignManagerRunCampaignFailed**

This event is sent when the Campaign Manager encounters an unexpected error while running the campaign

The following fields are available:

- **CampaignConfigVersion** Configuration version for the current campaign

- **CampaignID** Currently campaign that's running on UNP
- **ConfigCatalogVersion** Current catalog version of UNP
- **ContentVersion** Content version for the current campaign on UNP
- **CV** Correlation vector
- **DetectorVersion** Most recently run detector version for the current campaign on UNP
- **GlobalEventCounter** Client-side counter that indicates the event ordering sent by the user
- **hresult** HRESULT of the failure#N#
- **PackageVersion** Current UNP package version

## Upgrade events

### Setup360Telemetry.PreDownloadUX

The event sends data regarding OS updates and upgrades from Windows 7, Windows 8, and Windows 10. Specifically, the Setup360Telemetry.PredownloadUX indicates the outcome of the PredownloadUX portion of the update process.

The following fields are available:

- **ClientId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **HostOSBuildNumber** The build number of the previous operating system.
- **HostOsSkuName** The OS edition which is running the Setup360 instance (previous operating system).
- **InstanceId** Unique GUID that identifies each instance of setuphost.exe.
- **ReportId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that can be used to diagnose errors.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **SetupVersionBuildNumber** The build number of Setup360 (build number of the target OS).
- **State** The exit state of the Setup360 run. Example: succeeded, failed, blocked, cancelled
- **TestId** A string to uniquely identify a group of events.
- **Wuid** Windows Update client ID.

### Setup360Telemetry.UnexpectedEvent

This event sends data indicating that the device has invoked the unexpected event phase of the upgrade, to help keep Windows up to date.

The following fields are available:

- **ClientId** With Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **HostOSBuildNumber** The build number of the previous OS.
- **HostOsSkuName** The OS edition which is running Setup360 instance (previous OS).
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe
- **ReportId** With Windows Update, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback



- **Setup360Result** The result of Setup360. This is an HRESULT error code that can be used to diagnose errors.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **TestId** A string to uniquely identify a group of events.
- **Wuld** This is the Windows Update Client ID. With Windows Update, this is the same as the clientId.

#### Setup360Telemetry.PreInstallQuiet

This event sends data indicating that the device has invoked the preinstall quiet phase of the upgrade, to help keep Windows up to date.

The following fields are available:

- **ClientId** With Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **HostOSBuildNumber** The build number of the previous OS.
- **HostOsSkuName** The OS edition which is running Setup360 instance (previous OS).
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe
- **ReportId** With Windows Update, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback etc.
- **Setup360Result** The result of Setup360. This is an HRESULT error code that can be used to diagnose errors.
- **Setup360Scenario** Setup360 flow type (Boot, Media, Update, MCT)
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **TestId** A string to uniquely identify a group of events.
- **Wuld** This is the Windows Update Client ID. With Windows Update, this is the same as the clientId.

#### Setup360Telemetry.Finalize

This event sends data indicating that the device has invoked the finalize phase of the upgrade, to help keep Windows up-to-date.

The following fields are available:

- **ClientId** With Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **HostOSBuildNumber** The build number of the previous OS.
- **HostOsSkuName** The OS edition which is running Setup360 instance (previous OS).
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe
- **ReportId** With Windows Update, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that is used to diagnose errors.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **TestId** A string to uniquely identify a group of events.

- **Wuld** This is the Windows Update Client ID. With Windows Update, this is the same as the clientId.

### Setup360Telemetry.PostRebootInstall

This event sends data indicating that the device has invoked the postrebootinstall phase of the upgrade, to help keep Windows up-to-date.

The following fields are available:

- **ClientId** With Windows Update, this is the Windows Update client ID that is passed to Setup. In Media setup, the default value is Media360, but can be overwritten by the caller to a unique value.
- **HostOSBuildNumber** The build number of the previous OS.
- **HostOsSkuName** The OS edition which is running Setup360 instance (previous OS).
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe.
- **ReportId** With Windows Update, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that's used to diagnose errors.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **TestId** A string to uniquely identify a group of events.
- **Wuld** This is the Windows Update Client ID. With Windows Update, this is the same as ClientId.

### Setup360Telemetry.PreDownloadQuiet

This event sends data indicating that the device has invoked the predownload quiet phase of the upgrade, to help keep Windows up to date.

The following fields are available:

- **ClientId** Using Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **HostOSBuildNumber** The build number of the previous OS.
- **HostOsSkuName** The OS edition which is running Setup360 instance (previous operating system).
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe
- **ReportId** Using Windows Update, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that is used to diagnose errors.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, canceled
- **TestId** A string to uniquely identify a group of events.
- **Wuld** This is the Windows Update Client ID. Using Windows Update, this is the same as the clientId.

### Setup360Telemetry.OsUninstall

The event sends data regarding OS updates and upgrades from Windows 7, Windows 8, and Windows 10. Specifically, the Setup360Telemetry.OSUninstall indicates the outcome of an OS uninstall.

The following fields are available:

- **ClientId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **HostOSBuildNumber** The build number of the previous OS.
- **HostOsSkuName** The OS edition which is running the Setup360 instance (previous OS).
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe.
- **ReportId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that is used to diagnose errors.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).
- **State** Exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **TestId** A string to uniquely identify a group of events.
- **Wuld** Windows Update client ID.

#### Setup360Telemetry.Downlevel

This event sends data indicating that the device has invoked the downlevel phase of the upgrade. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **ClientId** If using Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, the default value is Media360, but it can be overwritten by the caller to a unique value.
- **HostOSBuildNumber** The build number of the downlevel OS.
- **HostOsSkuName** The operating system edition which is running Setup360 instance (downlevel OS).
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe.
- **ReportId** In the Windows Update scenario, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. It's an HRESULT error code that can be used to diagnose errors.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **SetupVersionBuildNumber** The build number of Setup360 (build number of the target OS).
- **State** Exit state of given Setup360 run. Example: succeeded, failed, blocked, cancelled
- **TestId** A string that uniquely identifies a group of events.
- **Wuld** This is the Windows Update Client ID. In the Windows Update scenario, this is the same as the clientId.

#### Setup360Telemetry.PreInstallUX

This event sends data regarding OS updates and upgrades from Windows 7, Windows 8, and Windows 10. Specifically, the Setup360Telemetry.PreinstallUX indicates the outcome of the PreinstallUX portion of the update process.

The following fields are available:

- **ClientId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **HostOSBuildNumber** The build number of the previous OS.

- **HostOsSkuName** The OS edition which is running the Setup360 instance (previous OS).
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe.
- **ReportId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that is used to diagnose errors.
- **Setup360Scenario** The Setup360 flow type, Example: Boot, Media, Update, MCT
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **TestId** A string to uniquely identify a group of events.
- **Wuld** Windows Update client ID.

### Setup360Telemetry.Setup360

This event sends data about OS deployment scenarios, to help keep Windows up-to-date.

The following fields are available:

- **FieldName** Retrieves the data point.
- **FlightData** Specifies a unique identifier for each group of Windows Insider builds.
- **InstanceId** Retrieves a unique identifier for each instance of a setup session.
- **ReportId** Retrieves the report ID.
- **ScenarioId** Retrieves the deployment scenario.
- **Value** Retrieves the value associated with the corresponding FieldName.
- **ClientId** Retrieves the upgrade ID: Upgrades via Windows Update - specifies the WU clientID. All other deployment - static string.

## Windows as a Service diagnostic events

### Microsoft.Windows.WaaSMedic.SummaryEvent

This event provides the results from the WaaSMedic engine

The following fields are available:

- **detectionSummary** Result of each detection that ran
- **featureAssessmentImpact** Windows as a Service (WaaS) Assessment impact on feature updates
- **insufficientSessions** True, if the device has enough activity to be eligible for update diagnostics. False, if otherwise
- **isManaged** Indicates the device is managed for updates
- **isWUConnected** Indicates the device is connected to Windows Update
- **noMoreActions** All available WaaSMedic diagnostics have run. There are no pending diagnostics and corresponding actions
- **qualityAssessmentImpact** Windows as a Service (WaaS) Assessment impact for quality updates
- **remediationSummary** Result of each operation performed on a device to fix an invalid state or configuration that's preventing the device from getting updates. For example, if Windows Update service is turned off, the fix is to turn the it back on
- **usingBackupFeatureAssessment** The WaaSMedic engine contacts Windows as a Service (WaaS) Assessment to determine whether the device is up-to-date. If WaaS Assessment isn't available, the engine falls back to backup feature assessments, which are determined programmatically on the client#N#
- **usingBackupQualityAssessment** The WaaSMedic engine contacts Windows as a Service (WaaS)

Assessment to determine whether the device is up-to-date. If WaaS Assessment isn't available, the engine falls back to backup quality assessments, which are determined programmatically on the client#N#

- **versionString** Installed version of the WaaS Medic engine
- **hrEngineResult** Indicates the WaaS Medic engine operation error codes

### Microsoft.Windows.WaaS.Medic.Summary

This event provides the results of the WaaS Medic diagnostic run

The following fields are available:

- **detectionSummary** Result of each detection that ran
- **remediationSummary** Result of each operation performed on a device to fix an invalid state or configuration that's preventing the device from getting updates. For example, if Windows Update service is turned off, the fix is to turn the it back on
- **versionString** Installed version of the WaaS Medic engine
- **featureAssessmentImpact** Windows as a Service (WaaS) Assessment impact on feature updates
- **insufficientSessions** True, if the device has enough activity to be eligible for update diagnostics. False, if otherwise
- **isManaged** Indicates the device is managed for updates
- **isWUConnected** Indicates the device is connected to Windows Update
- **noMoreActions** All available WaaS Medic diagnostics have run. There are no pending diagnostics and corresponding actions
- **qualityAssessmentImpact** Windows as a Service (WaaS) Assessment impact for quality updates
- **usingBackupFeatureAssessment** The WaaS Medic engine contacts Windows as a Service (WaaS) Assessment to determine whether the device is up-to-date. If WaaS Assessment isn't available, the engine falls back to backup feature assessments, which are determined programmatically on the client
- **usingBackupQualityAssessment** The WaaS Medic engine contacts Windows as a Service (WaaS) Assessment to determine whether the device is up-to-date. If WaaS Assessment isn't available, the engine falls back to backup quality assessments, which are determined programmatically on the client

## Windows Error Reporting events

### Microsoft.Windows.WERVertical.OSCrash

This event sends binary data from the collected dump file whenever a bug check occurs, to help keep Windows up to date. This is the OneCore version of this event.

The following fields are available:

- **BootId** UInt32 identifying the boot number for this device.
- **BugCheckCode** "UInt64 ""bugcheck code"" that identifies a proximate cause of the bug check."
- **BugCheckParameter1** UInt64 parameter providing additional information.
- **BugCheckParameter2** UInt64 parameter providing additional information.
- **BugCheckParameter3** UInt64 parameter providing additional information.
- **BugCheckParameter4** UInt64 parameter providing additional information.
- **DumpFileAttributes** Codes that identify the type of data contained in the dump file
- **DumpFileSize** Size of the dump file
- **IsValidDumpFile** True if the dump file is valid for the debugger, false otherwise
- **ReportId** WER Report Id associated with this bug check (used for finding the corresponding report archive in Watson).

## Microsoft Store events

### Microsoft.Windows.StoreAgent.Telemetry.AbortedInstallation

This event is sent when an installation or update is canceled by a user or the system and is used to help keep Windows Apps up to date and secure.

The following fields are available:

- **AggregatedPackageFullNames** The names of all packages to be downloaded and installed.
- **AttemptNumber** Number of retry attempts before it was canceled.
- **BundleId** The Item Bundle ID.
- **CategoryId** The Item Category ID.
- **ClientAppId** The identity of the app that initiated this operation.
- **HResult** The result code of the last action performed before this operation.
- **IntentPFNs** Intent Product Family Name
- **IsBundle** Is this a bundle?
- **IsInteractive** Was this requested by a user?
- **IsMandatory** Was this a mandatory update?
- **IsRemediation** Was this a remediation install?
- **IsRestore** Is this automatically restoring a previously acquired product?
- **IsUpdate** Flag indicating if this is an update.
- **IsWin32** Flag indicating if this is a Win32 app (not used).
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **PFN** The product family name of the product being installed.
- **ProductId** The identity of the package or packages being installed.
- **SystemAttemptNumber** The total number of automatic attempts at installation before it was canceled.
- **UpdateId** Update ID (if this is an update)
- **UserAttemptNumber** The total number of user attempts at installation before it was canceled.
- **WUContentId** The Windows Update content ID

### Microsoft.Windows.StoreAgent.Telemetry.EndAcquireLicense

This event is sent after the license is acquired when a product is being installed. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **AggregatedPackageFullNames** Includes a set of package full names for each app that is part of an atomic set.
- **AttemptNumber** The total number of attempts to acquire this product.
- **BundleId** The bundle ID
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **HResult** HRESULT code to show the result of the operation (success/failure).
- **IntentPFNs** Intent Product Family Name
- **IsBundle** Is this a bundle?
- **IsInteractive** Did the user initiate the installation?
- **IsMandatory** Is this a mandatory update?
- **IsRemediation** Is this repairing a previous installation?
- **IsRestore** Is this happening after a device restore?
- **IsUpdate** Is this an update?
- **IsWin32** Flag indicating if this is a Win32app.
- **ParentBundleId** The product's parent bundle ID.

- **ParentBundleId** The parent bundle ID (if it's part of a bundle).
- **PFN** Product Family Name of the product being installed.
- **ProductId** The Store Product ID for the product being installed.
- **SystemAttemptNumber** The number of attempts by the system to acquire this product.
- **UpdateId** The update ID (if this is an update)
- **UserAttemptNumber** The number of attempts by the user to acquire this product
- **WUContentId** The Windows Update content ID

### **Microsoft.Windows.StoreAgent.Telemetry.EndDownload**

This event happens during the app update or installation when content is being downloaded at the end of the process to report success or failure. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **AggregatedPackageFullNames** The name of all packages to be downloaded and installed.
- **AttemptNumber** Number of retry attempts before it was canceled.
- **BundleId** The identity of the Windows Insider build associated with this product.
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **DownloadSize** The total size of the download.
- **ExtendedHResult** Any extended HRESULT error codes.
- **HResult** The result code of the last action performed.
- **IntentPFNs** Intent Product Family Name
- **IsBundle** Is this a bundle?
- **IsInteractive** Is this initiated by the user?
- **IsMandatory** Is this a mandatory installation?
- **IsRemediation** Is this repairing a previous installation?
- **IsRestore** Is this a restore of a previously acquired product?
- **IsUpdate** Is this an update?
- **IsWin32** Flag indicating if this is a Win32 app (unused).
- **ParentBundleId** The parent bundle ID (if it's part of a bundle).
- **PFN** The Product Family Name of the app being download.
- **ProductId** The Store Product ID for the product being installed.
- **SystemAttemptNumber** The number of attempts by the system to download.
- **UpdateId** Update ID (if this is an update)
- **UserAttemptNumber** The number of attempts by the user to download.
- **WUContentId** The Windows Update content ID.

### **Microsoft.Windows.StoreAgent.Telemetry.EndFrameworkUpdate**

This event happens when an app update requires an updated Framework package and the process starts to download it. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **HResult** The result code of the last action performed before this operation.

### **Microsoft.Windows.StoreAgent.Telemetry.EndGetInstalledContentIds**

This event is sent after sending the inventory of the products installed to determine whether updates for those products are available. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **HResult** The result code of the last action performed before this operation.

### **Microsoft.Windows.StoreAgent.Telemetry.EndInstall**

This event is sent after a product has been installed. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **AggregatedPackageFullNames** The names of all packages to be downloaded and installed.
- **AttemptNumber** The number of retry attempts before it was canceled.
- **BundleId** The identity of the build associated with this product.
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **ExtendedHResult** The extended HResult error code.
- **HResult** The result code of the last action performed.
- **IntentPFNs** Intent Product Family Name
- **IsBundle** Is this a bundle?
- **IsInteractive** Is this an interactive installation?
- **IsMandatory** Is this a mandatory installation?
- **IsRemediation** Is this repairing a previous installation?
- **IsRestore** Is this automatically restoring a previously acquired product?
- **IsUpdate** Is this an update?
- **IsWin32** Flag indicating if this a Win32 app (unused).
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **PFN** Product Family Name of the product being installed.
- **ProductId** The Store Product ID for the product being installed.
- **SystemAttemptNumber** The total number of system attempts.
- **UpdateId** Update ID (if this is an update)
- **UserAttemptNumber** The total number of user attempts.
- **WUContentId** The Windows Update content ID

### **Microsoft.Windows.StoreAgent.Telemetry.EndScanForUpdates**

This event is sent after a scan for product updates to determine if there are packages to install. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **ClientAppId** The identity of the app that initiated this operation.
- **HResult** The result code of the last action performed.
- **IsApplicability** Is this request to only check if there are any applicable packages to install?
- **IsInteractive** Is this user requested?
- **IsOnline** Is the request doing an online check?

### **Microsoft.Windows.StoreAgent.Telemetry.EndSearchUpdatePackages**

This event is sent after searching for update packages to install. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **AggregatedPackageFullNames** The names of all packages to be downloaded and installed.
- **AttemptNumber** The total number of retry attempts before it was canceled.
- **BundleId** The identity of the build associated with this product.
- **CategoryId** The identity of the package or packages being installed.



- **ClientAppId** The identity of the app that initiated this operation.
- **HResult** The result code of the last action performed.
- **IntentPFNs** The licensing identity of this package.
- **IsBundle** Is this a bundle?
- **IsInteractive** Is this user requested?
- **IsMandatory** Is this a mandatory update?
- **IsRemediation** Is this repairing a previous installation?
- **IsRestore** Is this restoring previously acquired content?
- **IsUpdate** Is this an update?
- **IsWin32** Flag indicating if this a Win32 app (unused).
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **PFN** The name of the package or packages requested for install.
- **ProductId** The Store Product ID for the product being installed.
- **SystemAttemptNumber** The total number of system attempts.
- **UpdateId** Update ID (if this is an update)
- **UserAttemptNumber** The total number of user attempts.
- **WUContentId** The Windows Update content ID

#### **Microsoft.Windows.StoreAgent.Telemetry.EndStageUserData**

This event is sent between download and installation to see if there is app data that needs to be restored from the cloud. It's used to keep Windows up-to-date and secure.

The following fields are available:

- **AttemptNumber** The total number of retry attempts before it was canceled.
- **BundleId** The identity of the build associated with this product.
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **HResult** The result code of the last action performed.
- **IsBundle** Is this a bundle?
- **IsInteractive** Is this user requested?
- **IsMandatory** Is this a mandatory update?
- **IsRemediation** Is this repairing a previous installation?
- **IsRestore** Is this restoring previously acquired content?
- **IsUpdate** Is this an update?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **PFN** The name of the package or packages requested for install.
- **ProductId** The Store Product ID for the product being installed.
- **SystemAttemptNumber** The total number of system attempts.
- **UserAttemptNumber** The total number of system attempts.
- **WUContentId** The Windows Update content ID
- **IntentPFNs** The licensing identity of this package.
- **AggregatedPackageFullNames** The name of all packages to be downloaded and installed.

#### **Microsoft.Windows.StoreAgent.Telemetry.InstallOperationRequest**

This event happens at the beginning of the install process when an app update or new app is installed. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **BundleId** The identity of the build associated with this product.
- **CatalogId** If this product is from a private catalog, the Store Product ID for the product being installed.
- **ProductId** The Store Product ID for the product being installed.
- **Skuid** Specific edition ID being installed.
- **VolumePath** The disk path of the installation.

#### Microsoft.Windows.StoreAgent.Telemetry.PauseInstallation

This event is sent when a product install or update is paused either by a user or the system. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **AttemptNumber** The total number of retry attempts before it was canceled.
- **BundleId** The identity of the build associated with this product.
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **IsBundle** Is this a bundle?
- **IsInteractive** Is this user requested?
- **IsMandatory** Is this a mandatory update?
- **IsRemediation** Is this repairing a previous installation?
- **IsRestore** Is this restoring previously acquired content?
- **IsUpdate** Is this an update?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **PFN** The Product Full Name.
- **PreviousHResult** The result code of the last action performed before this operation.
- **PreviousInstallState** Previous state before the installation or update was paused.
- **ProductId** The Store Product ID for the product being installed.
- **RelatedCV** Correlation Vector of a previous performed action on this product.
- **SystemAttemptNumber** The total number of system attempts.
- **UserAttemptNumber** The total number of user attempts.
- **WUContentId** The Windows Update content ID
- **IntentPFNs** The licensing identity of this package.
- **AggregatedPackageFullNames** The names of all packages to be downloaded and installed.

#### Microsoft.Windows.StoreAgent.Telemetry.ResumeInstallation

This event happens when a product install or update is resumed either by a user or the system. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **AttemptNumber** The number of retry attempts before it was canceled.
- **BundleId** The identity of the build associated with this product.
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **IsBundle** Is this a bundle?
- **IsInteractive** Is this user requested?
- **IsMandatory** Is this a mandatory update?
- **IsRemediation** Is this repairing a previous installation?
- **IsRestore** Is this restoring previously acquired content?
- **IsUpdate** Is this an update?

- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **PFN** The name of the package or packages requested for install.
- **PreviousHResult** The previous HResult error code.
- **PreviousInstallState** Previous state before the installation was paused.
- **ProductId** The Store Product ID for the product being installed.
- **RelatedCV** Correlation Vector for the original install before it was resumed.
- **SystemAttemptNumber** The total number of system attempts.
- **UserAttemptNumber** The total number of user attempts.
- **WUContentId** The Windows Update content ID
- **IntentPFNs** Intent Product Family Name
- **AggregatedPackageFullNames** The names of all packages to be downloaded and installed.
- **HResult** The result code of the last action performed before this operation.
- **IsUserRetry** Did the user initiate the retry?

#### **Microsoft.Windows.StoreAgent.Telemetry.UpdateAppOperationRequest**

This event happens an app for a user needs to be updated. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **PFamN** The name of the product that is requested for update.

#### **Microsoft.Windows.StoreAgent.Telemetry.CancelInstallation**

This event is sent when an app update or installation is canceled while in interactive mode. This can be canceled by the user or the system. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **AttemptNumber** Total number of installation attempts.
- **BundleId** The identity of the Windows Insider build that is associated with this product.
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **IsBundle** Is this a bundle?
- **IsInteractive** Was this requested by a user?
- **IsMandatory** Is this a mandatory update?
- **IsRemediation** Is this repairing a previous installation?
- **IsRestore** Is this an automatic restore of a previously acquired product?
- **IsUpdate** Is this a product update?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **PFN** The name of all packages to be downloaded and installed.
- **PreviousHResult** The previous HResult code.
- **PreviousInstallState** Previous installation state before it was canceled.
- **ProductId** The name of the package or packages requested for installation.
- **RelatedCV** Correlation Vector of a previous performed action on this product.
- **SystemAttemptNumber** Total number of automatic attempts to install before it was canceled.
- **UserAttemptNumber** Total number of user attempts to install before it was canceled.
- **WUContentId** The Windows Update content ID
- **IntentPFNs** Intent Product Family Name
- **AggregatedPackageFullNames** The names of all package or packages to be downloaded and installed.

#### **Microsoft.Windows.StoreAgent.Telemetry.SearchForUpdateOperationRequest**

This event is sent when searching for update packages to install. It's used to help keep Windows up-to-date and

secure.

The following fields are available:

- **CatalogId** The Store Product ID for the product being installed.
- **ProductId** The Store Product ID for the product being installed.
- **Skuld** Specific edition of the app being updated.

#### **Microsoft.Windows.StoreAgent.Telemetry.EndUpdateMetadataPrepare**

This event happens after a scan for available app updates. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **HResult** The result code of the last action performed.

#### **Microsoft.Windows.StoreAgent.Telemetry.CompleteInstallOperationRequest**

This event is sent after the app installations or updates. It's used to help keep Windows up-to-date and secure

The following fields are available:

- **CatalogId** The Store Product ID of the app being installed.
- **HResult** HRESULT code of the action being performed.
- **IsBundle** Is this a bundle?
- **PackageFamilyName** The name of the package being installed.
- **ProductId** The Store Product ID of the product being installed.
- **Skuld** Specific edition of the item being installed.

#### **Microsoft.Windows.StoreAgent.Telemetry.ResumeOperationRequest**

This event happens when a product install or update is resumed by a user and on installation retries. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **ProductId** The Store Product ID for the product being installed.

#### **Microsoft.Windows.StoreAgent.Telemetry.FulfillmentComplete**

This event is sent at the end of an app install or update and is used to track the very end of the install or update process.

The following fields are available:

- **FailedRetry** Was the installation or update retry successful?
- **HResult** The HRESULT code of the operation.
- **PFN** The Package Family Name of the app that is being installed or updated.
- **ProductId** The product ID of the app that is being updated or installed.

#### **Microsoft.Windows.StoreAgent.Telemetry.FulfillmentInitiate**

This event is sent at the beginning of an app install or update and is used to track the very beginning of the install or update process.

The following fields are available:

- **PFN** The Package Family Name of the app that is being installed or updated.
- **ProductId** The product ID of the app that is being updated or installed.

## Windows Update Delivery Optimization events

## Microsoft.OSG.DU.DeliveryOptClient.DownloadCompleted

This event describes when a download has completed with Delivery Optimization. It's used to understand and address problems regarding downloads.

The following fields are available:

- **background** Is the download a background download?
- **bytesFromCDN** The number of bytes received from a CDN source.
- **bytesFromGroupPeers** The number of bytes received from a peer in the same domain group.
- **bytesFromIntPeers** The number of bytes received from peers not in the same LAN or in the same domain group.
- **bytesFromPeers** The number of bytes received from a peer in the same LAN.
- **bytesRequested** The total number of bytes requested for download.
- **cdnConnectionCount** The total number of connections made to the CDN.
- **cdnErrorCodes** A list of CDN connection errors since the last FailureCDNCommunication event.
- **cdnErrorCounts** The number of times each error in cdnErrorCodes was encountered.
- **cdnIp** The IP address of the source CDN.
- **clientTelId** A random number used for device sampling.
- **doErrorCode** The Delivery Optimization error code that was returned.
- **downloadBps** The maximum measured available download bandwidth (in bytes per second).
- **downloadUsageBps** The download speed (in bytes per second).
- **downloadMode** The download mode used for this file download session.
- **fileID** The ID of the file being downloaded.
- **fileSize** The size of the file being downloaded.
- **groupConnectionCount** The total number of connections made to peers in the same group.
- **internetConnectionCount** The total number of connections made to peers not in the same LAN or the same group.
- **lanConnectionCount** The total number of connections made to peers in the same LAN.
- **numPeers** The total number of peers used for this download.
- **restrictedUpload** Is the upload restricted?
- **scenarioID** The ID of the scenario.
- **sessionID** The ID of the download session.
- **totalTimeMs** Duration of the download (in seconds).
- **updateID** The ID of the update being downloaded.
- **uplinkBps** The maximum measured available upload bandwidth (in bytes per second).
- **uplinkUsageBps** The upload speed (in bytes per second).
- **experimentId** When running a test, this is used to correlate with other events that are part of the same test.
- **isVpn** Is the device connected to a Virtual Private Network?
- **usedMemoryStream** Did the download use memory streaming?

## Microsoft.OSG.DU.DeliveryOptClient.DownloadPaused

This event represents a temporary suspension of a download with Delivery Optimization. It's used to understand and address problems regarding downloads.

The following fields are available:

- **background** Is the download a background download?
- **clientTelId** A random number used for device sampling.
- **errorCode** The error code that was returned.
- **fileID** The ID of the file being paused.

- **reasonCode** The reason for pausing the download.
- **scenarioID** The ID of the scenario.
- **sessionID** The ID of the download session.
- **updateID** The ID of the update being paused.
- **experimentId** When running a test, this is used to correlate with other events that are part of the same test.
- **isVpn** Is the device connected to a Virtual Private Network?

#### **Microsoft.OSG.DU.DeliveryOptClient.JobError**

This event represents a Windows Update job error. It allows for investigation of top errors.

The following fields are available:

- **clientTelId** A random number used for device sampling.
- **errorCode** The error code returned.
- **experimentId** When running a test, this is used to correlate with other events that are part of the same test.
- **fileID** The ID of the file being downloaded.
- **jobID** The Windows Update job ID.

#### **Microsoft.OSG.DU.DeliveryOptClient.DownloadCanceled**

This event describes when a download was canceled with Delivery Optimization. It's used to understand and address problems regarding downloads.

The following fields are available:

- **background** Is the download being done in the background?
- **bytesFromCDN** The number of bytes received from a CDN source.
- **bytesFromGroupPeers** The number of bytes received from a peer in the same group.
- **bytesFromIntPeers** The number of bytes received from peers not in the same LAN or in the same group.
- **bytesFromPeers** The number of bytes received from a peer in the same LAN.
- **cdnErrorCodes** A list of CDN connection errors since the last FailureCDNCommunication event.
- **cdnErrorCounts** The number of times each error in cdnErrorCodes was encountered.
- **clientTelId** A random number used for device sampling.
- **doErrorCode** The Delivery Optimization error code that was returned.
- **errorCode** The error code that was returned.
- **experimentId** When running a test, this is used to correlate events that are part of the same test.
- **fileID** The ID of the file being downloaded.
- **isVpn** Is the device connected to a Virtual Private Network?
- **scenarioID** The ID of the scenario.
- **sessionID** The ID of the file download session.
- **updateID** The ID of the update being downloaded.
- **usedMemoryStream** Did the download use memory streaming?

#### **Microsoft.OSG.DU.DeliveryOptClient.DownloadStarted**

This event describes the start of a new download with Delivery Optimization. It's used to understand and address problems regarding downloads.

The following fields are available:

- **background** Is the download a background download?
- **cdnUrl** The URL of the CDN.
- **clientTelId** A random number used for device sampling.
- **deviceProfile** Identifies the usage or form factor. Example: Desktop or Xbox

- **diceRoll** The dice roll value used in sampling events.
- **doClientVersion** The version of the Delivery Optimization client.
- **doErrorCode** The Delivery Optimization error code that was returned.
- **downloadMode** The download mode used for this file download session.
- **errorCode** The error code that was returned.
- **experimentId** When running a test, this is used to correlate with other events that are part of the same test.
- **fileId** The ID of the file being downloaded.
- **filePath** The path where the file will be written.
- **groupId** ID for the group.
- **isVpn** Is the device connected to a Virtual Private Network?
- **jobId** The ID of the Windows Update job.
- **minDiskSizeGB** The minimum disk size (in GB) required for Peering.
- **minDiskSizePolicyEnforced** Is the minimum disk size enforced via policy?
- **minFileSizePolicy** The minimum content file size policy to allow the download using Peering.
- **peerId** The ID for this Delivery Optimization client.
- **scenarioId** The ID of the scenario.
- **sessionId** The ID of the download session.
- **updateId** The ID of the update being downloaded.
- **usedMemoryStream** Did the download use memory streaming?
- **costFlags** A set of flags representing network cost.

#### **Microsoft.OSG.DU.DeliveryOptClient.FailureCdnCommunication**

This event represents a failure to download from a CDN with Delivery Optimization. It's used to understand and address problems regarding downloads.

The following fields are available:

- **cdnIp** The IP address of the CDN.
- **cdnUrl** The URL of the CDN.
- **clientTelId** A random number used for device sampling.
- **errorCode** The error code that was returned.
- **errorCount** The total number of times this error code was seen since the last FailureCdnCommunication event was encountered.
- **httpStatusCode** The HTTP status code returned by the CDN.
- **sessionId** The ID of the download session.
- **cdnHeaders** The HTTP headers returned by the CDN.
- **experimentId** When running a test, this is used to correlate with other events that are part of the same test.
- **fileId** The ID of the file being downloaded.
- **isHeadRequest** The type of HTTP request that was sent to the CDN. Example: HEAD or GET
- **requestSize** The size of the range requested from the CDN.
- **responseSize** The size of the range response received from the CDN.

## Windows Update events

#### **Microsoft.Windows.Update.DeviceUpdateAgent.UpdateAgentModeStart**

This event sends data for the start of each mode during the process of updating device manifest assets via the UUP (Unified Update Platform) update scenario, which is used to install a device manifest describing a set of driver packages.

The following fields are available:

- **flightId** The unique identifier for each flight
- **mode** Indicates that the Update Agent mode that has started. 1 = Initialize, 2 = DownloadRequest, 3 = Install, 4 = Commit
- **objectId** Unique value for each Update Agent mode
- **relatedCV** Correlation vector value generated from the latest scan
- **scenarioId** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **sessionId** Unique value for each Update Agent mode attempt
- **updateId** Unique ID for each update

#### **Microsoft.Windows.Update.DeviceUpdateAgent.UpdateAgentInitialize**

This event sends data for initializing a new update session for the new device manifest UUP (Unified Update Platform) update scenario, which is used to install a device manifest describing a set of driver packages

The following fields are available:

- **errorCode** The error code returned for the current initialize phase
- **flightId** The unique identifier for each flight
- **flightMetadata** Contains the FlightId and the build being flighted
- **objectId** Unique value for each Update Agent mode
- **relatedCV** Correlation vector value generated from the latest USO scan
- **result** Result of the initialize phase of update. 0 = Succeeded, 1 = Failed, 2 = Cancelled, 3 = Blocked, 4 = BlockCancelled
- **scenarioId** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate#N#
- **sessionData** Contains instructions to update agent for processing FODs and DUICs (Null for other scenarios)
- **sessionId** "Unique value for each Update Agent mode attempt "
- **updateId** Unique ID for each update

#### **Microsoft.Windows.Update.DeviceUpdateAgent.UpdateAgentCommit**

This event collects information regarding the final commit phase of the new device manifest UUP (Unified Update Platform) update scenario, which is used to install a device manifest describing a set of driver packages

The following fields are available:

- **errorCode** The error code returned for the current session initialization
- **flightId** The unique identifier for each flight
- **objectId** The unique GUID for each diagnostics session
- **relatedCV** A correlation vector value, generated from the latest USO scan
- **result** Outcome of the initialization of the session
- **scenarioId** Identifies the Update scenario
- **sessionId** The unique value for each update session
- **updateId** The unique identifier for each Update

#### **Microsoft.Windows.Update.DeviceUpdateAgent.UpdateAgentInstall**

This event collects information regarding the install phase of the new device manifest UUP (Unified Update Platform) update scenario, which is used to install a device manifest describing a set of driver packages

The following fields are available:

- **errorCode** The error code returned for the current install phase
- **flightId** The unique identifier for each flight



- **objectId** Unique value for each Update Agent mode
- **relatedCV** Correlation vector value generated from the latest scan
- **result** Result of the install phase of update. 0 = Succeeded 1 = Failed, 2 = Cancelled, 3 = Blocked, 4 = BlockCancelled
- **scenarioId** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **sessionId** Unique value for each Update Agent mode attempt
- **updateId** Unique ID for each update

#### **Microsoft.Windows.Update.DeviceUpdateAgent.UpdateAgentDownloadRequest**

This event collects information regarding the download request phase of the new device manifest UUP (Unified Update Platform) update scenario, which is used to install a device manifest describing a set of driver packages

The following fields are available:

- **deletedCorruptFiles** Indicates if UpdateAgent found any corrupt payload files and whether the payload was deleted
- **errorCode** The error code returned for the current session initialization
- **flightId** The unique identifier for each flight
- **objectId** Unique value for each Update Agent mode
- **packageCountOptional** Number of optional packages requested
- **packageCountRequired** Number of required packages requested
- **packageCountTotal** Total number of packages needed
- **packageCountTotalCanonical** Total number of canonical packages
- **packageCountTotalDiff** Total number of diff packages
- **packageCountTotalExpress** Total number of express packages
- **packageSizeCanonical** Size of canonical packages in bytes
- **packageSizeDiff** Size of diff packages in bytes
- **packageSizeExpress** Size of express packages in bytes
- **rangeRequestState** Represents the state of the download range request
- **relatedCV** Correlation vector value generated from the latest USO scan
- **result** Result of the download request phase of update
- **scenarioId** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **sessionId** Unique value for each Update Agent mode attempt
- **updateId** Unique ID for each update

#### **Microsoft.Windows.Update.Orchestrator.GameActive**

This event indicates that an enabled GameMode process prevented the device from restarting to complete an update

The following fields are available:

- **eventScenario** Indicates the purpose of sending this event - whether because the software distribution just started checking for content, or whether it was cancelled, succeeded, or failed
- **gameModeReason** Name of the enabled GameMode process that prevented the device from restarting to complete an update
- **wuDeviceId** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue

#### **Microsoft.Windows.Update.DataMigrationFramework.DmfMigrationCompleted**

This event sends data collected at the end of the Data Migration Framework (DMF) and parameters involved in its invocation, to help keep Windows up to date.

The following fields are available:

- **MigrationDurationInMilliseconds** How long the DMF migration took (in milliseconds)
- **MigrationEndTime** A system timestamp of when the DMF migration completed.
- **RevisionNumbers** A collection of revision numbers for the updates associated with the DMF session.
- **UpdateIds** A collection of GUIDs for updates that are associated with the DMF session.
- **WuClientId** The GUID of the Windows Update client responsible for triggering the DMF migration

#### **Microsoft.Windows.Update.DataMigrationFramework.DmfMigrationStarted**

This event sends data collected at the beginning of the Data Migration Framework (DMF) and parameters involved in its invocation, to help keep Windows up to date.

The following fields are available:

- **MigrationMicrosoftPhases** Revision numbers for the updates that were installed.
- **MigrationOEMPhases** WU Update IDs for the updates that were installed.
- **MigrationStartTime** The timestamp representing the beginning of the DMF migration
- **WuClientId** The GUID of the Windows Update client invoking DMF
- **RevisionNumbers** A collection of the revision numbers associated with the UpdateIds.
- **UpdateIds** A collection of GUIDs identifying the upgrades that are running.

#### **Microsoft.Windows.Update.DataMigrationFramework.MigratorResult**

This event sends DMF migrator data to help keep Windows up to date.

The following fields are available:

- **CurrentStep** This is the last step the migrator reported before returning a result. This tells us how far through the individual migrator the device was before failure.
- **ErrorCode** The result (as an HRESULT) of the migrator that just completed.
- **MigratorId** A GUID identifying the migrator that just completed.
- **MigratorName** The name of the migrator that just completed.
- **RunDurationInSeconds** The time it took for the migrator to complete.
- **TotalSteps** Migrators report progress in number of completed steps against the total steps. This is the total number of steps.

#### **Microsoft.Windows.Update.Orchestrator.Download**

This event sends launch data for a Windows Update download to help keep Windows up to date.

The following fields are available:

- **deferReason** Reason for download not completing
- **detectionDeferreason** Reason for download not completing
- **errorCode** An error code represented as a hexadecimal value
- **eventScenario** End to end update session ID.
- **flightID** Unique update ID.
- **interactive** Identifies if session is user initiated.
- **revisionNumber** Update revision number.
- **updateId** Update ID.
- **updateScenarioType** The update session type.
- **wuDeviceid** Unique device ID used by Windows Update.

### Microsoft.Windows.Update.Orchestrator.FlightInapplicable

This event sends data on whether the update was applicable to the device, to help keep Windows up to date.

The following fields are available:

- **EventPublishedTime** time that the event was generated
- **revisionNumber** Revision Number of the Update
- **updateId** Unique Update ID
- **UpdateStatus** Integer that describes Update state
- **wuDeviceid** Unique Device ID
- **flightID** Unique Update ID
- **updateScenarioType** The update session type.

### Microsoft.Windows.Update.Orchestrator.PostInstall

This event sends data about lite stack devices (mobile, IOT, anything non-PC) immediately before data migration is launched to help keep Windows up to date.

The following fields are available:

- **batteryLevel** Current battery capacity in mWh or percentage left.
- **bundleId** Update grouping ID.
- **bundleRevisionnumber** Bundle revision number.
- **errorCode** Hex code for the error message, to allow lookup of the specific error.
- **eventScenario** End to end update session ID.
- **flightID** Unique update ID.
- **sessionType** Interactive vs. Background.
- **wuDeviceid** Unique device ID used by Windows Update.

### Microsoft.Windows.Update.Orchestrator.RebootFailed

This event sends information about whether an update required a reboot and reasons for failure to help keep Windows up to date.

The following fields are available:

- **batteryLevel** Current battery capacity in mWh or percentage left.
- **deferReason** Reason for install not completing.
- **EventPublishedTime** The time that the reboot failure occurred.
- **flightID** Unique update ID.
- **installRebootDeferreason** Reason for reboot not occurring.
- **rebootOutsideOfActiveHours** Indicates the timing that the reboot was to occur to ensure the correct update process and experience is provided to keep Windows up to date.
- **RebootResults** Hex code indicating failure reason. Typically, we expect this to be a specific USO generated hex code.
- **revisionNumber** Update revision number.
- **updateId** Update ID.
- **updateScenarioType** The update session type.
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date.
- **wuDeviceid** Unique device ID used by Windows Update.

### Microsoft.Windows.Update.Orchestrator.UpdatePolicyCacheRefresh

This event sends data on whether Update Management Policies were enabled on a device, to help keep Windows

up to date.

The following fields are available:

- **configuredPoliciescount** Policy Count
- **policiesNamevaluesource** Policy Name
- **policyCacherefreshtime** Refresh time
- **updateInstalluxsetting** This shows whether a user has set policies via UX option
- **wuDeviceid** Unique device ID used by Windows Update.

#### **Microsoft.Windows.Update.Orchestrator.UpdateRebootRequired**

This event sends data about whether an update required a reboot to help keep Windows up to date.

The following fields are available:

- **revisionNumber** Update revision number.
- **updateId** Update ID.
- **wuDeviceid** Unique device ID used by Windows Update.
- **flightID** Unique update ID.
- **interactive** Indicates the reboot initiation stage of the update process was entered as a result of user action or not.
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date.
- **updateScenarioType** The update session type.

#### **Microsoft.Windows.Update.Ux.MusNotification.RebootScheduled**

This event sends data about a required reboot that is scheduled with no user interaction, to help keep Windows up to date.

The following fields are available:

- **activeHoursApplicable** True, If Active Hours applicable on this device. False, otherwise.
- **forcedReboot** True, if a reboot is forced on the device. Otherwise, this is False
- **rebootArgument** Argument for the reboot task. It also represents specific reboot related action.
- **rebootOutsideOfActiveHours** True, if a reboot is scheduled outside of active hours. False, otherwise.
- **rebootScheduledByUser** True, if a reboot is scheduled by user. False, if a reboot is scheduled automatically.
- **revisionNumber** Revision number of the update that is getting installed with this reboot.
- **scheduledRebootTime** Time of the scheduled reboot
- **updateId** Update ID of the update that is getting installed with this reboot.
- **wuDeviceid** Unique device ID used by Windows Update.
- **rebootState** The state of the reboot.

#### **Microsoft.Windows.Update.Orchestrator.Detection**

This event sends launch data for a Windows Update scan to help keep Windows up to date.

The following fields are available:

- **deferReason** Reason why the device could not check for updates.
- **detectionBlockreason** Reason for detection not completing.
- **detectionDeferreason** A log of deferral reasons for every update state.
- **errorCode** The returned error code.
- **eventScenario** End to end update session ID, or indicates the purpose of sending this event - whether because the software distribution just started installing content, or whether it was cancelled, succeeded, or failed.

- **flightID** A unique update ID.
- **interactive** Identifies if session is User Initiated.
- **revisionNumber** Update revision number.
- **updateId** Update ID.
- **updateScenarioType** The update session type.
- **wuDeviceid** Unique device ID used by Windows Update.

#### **Microsoft.Windows.Update.Orchestrator.InitiatingReboot**

This event sends data about an Orchestrator requesting a reboot from power management to help keep Windows up to date.

The following fields are available:

- **EventPublishedTime** Time of the event.
- **revisionNumber** Revision number of the update.
- **updateId** Update ID.
- **wuDeviceid** Unique device ID used by Windows Update.
- **flightID** Unique update ID
- **interactive** Indicates the reboot initiation stage of the update process was entered as a result of user action or not.
- **rebootOutsideOfActiveHours** Indicates the timing that the reboot was to occur to ensure the correct update process and experience is provided to keep Windows up to date.
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date.
- **updateScenarioType** The update session type.

#### **Microsoft.Windows.Update.Ux.MusUpdateSettings.RebootScheduled**

This event sends basic information for scheduling a device restart to install security updates. It's used to help keep Windows up-to-date.

The following fields are available:

- **activeHoursApplicable** Is the restart respecting Active Hours?
- **rebootArgument** The arguments that are passed to the OS for the restarted.
- **rebootOutsideOfActiveHours** Was the restart scheduled outside of Active Hours?
- **rebootScheduledByUser** Was the restart scheduled by the user? If the value is false, the restart was scheduled by the device.
- **rebootState** The state of the restart.
- **revisionNumber** The revision number of the OS being updated.
- **scheduledRebootTime** Time of the scheduled reboot
- **updateId** The Windows Update device GUID.
- **wuDeviceid** The Windows Update device GUID.
- **forcedReboot** True, if a reboot is forced on the device. Otherwise, this is False

#### **Microsoft.Windows.Update.Ux.MusNotification.RebootNoLongerNeeded**

This event is sent when a security update has successfully completed.

The following fields are available:

- **UtcTime** The Coordinated Universal Time that the restart was no longer needed.

#### **Microsoft.Windows.Update.Ux.MusNotification.ToastDisplayedToScheduleReboot**

This event is sent when a toast notification is shown to the user about scheduling a device restart.

The following fields are available:

- **UtcTime** The Coordinated Universal Time when the toast notification was shown.

#### **Microsoft.Windows.Update.Orchestrator.RestoreRebootTask**

This event sends data indicating that a reboot task is missing unexpectedly on a device and the task is restored because a reboot is still required, to help keep Windows up to date.

The following fields are available:

- **RebootTaskRestoredTime** Time at which this reboot task was restored.
- **revisionNumber** Update revision number.
- **updateId** Update ID.
- **wuDeviceid** Device id on which the reboot is restored

#### **Microsoft.Windows.Update.Orchestrator.SystemNeeded**

This event sends data about why a device is unable to reboot, to help keep Windows up to date.

The following fields are available:

- **eventScenario** End to end update session ID.
- **revisionNumber** Update revision number.
- **systemNeededReason** Reason ID
- **updateId** Update ID.
- **wuDeviceid** Unique device ID used by Windows Update.
- **rebootOutsideOfActiveHours** Indicates the timing that the reboot was to occur to ensure the correct update process and experience is provided to keep Windows up to date.
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date.
- **updateScenarioType** The update session type.

#### **Microsoft.Windows.Update.UpdateStackServicing.CheckForUpdates**

This event sends data about the UpdateStackServicing check for updates, to help keep Windows up to date.

The following fields are available:

- **BspVersion** The version of the BSP.
- **CallerApplicationName** The name of the USS scheduled task. Example UssScheduled or UssBoot
- **ClientVersion** The version of the client.
- **CommercializationOperator** The name of the operator.
- **DetectionVersion** The string returned from the GetDetectionVersion export of the downloaded detection DLL.
- **DeviceName** The name of the device.
- **EventInstanceId** The USS session ID.
- **EventScenario** The scenario of the event. Example: Started, Failed, or Succeeded
- **OemName** The name of the manufacturer.
- **ServiceGuid** The GUID of the service.
- **StatusCode** The HRESULT code of the operation.
- **WUDeviceID** The Windows Update device ID.

#### **Microsoft.Windows.Update.Orchestrator.CommitFailed**

This events tracks when a device needs to restart after an update but did not.

The following fields are available:

- **errorCode** The error code that was returned.
- **wuDeviceid** The Windows Update device GUID.

### **Microsoft.Windows.Update.Orchestrator.Install**

This event sends launch data for a Windows Update install to help keep Windows up to date.

The following fields are available:

- **batteryLevel** Current battery capacity in mWh or percentage left.
- **deferReason** Reason for install not completing.
- **eventScenario** End to end update session ID.
- **interactive** Identifies if session is user initiated.
- **wuDeviceid** Unique device ID used by Windows Update.
- **flightUpdate** Flight update
- **installRebootinitiatetime** The time it took for a reboot to be attempted.
- **minutesToCommit** The time it took to install updates.
- **revisionNumber** Update revision number.
- **updateId** Update ID.
- **errorCode** The error code represented by a hexadecimal value.
- **installCommitfailedtime** The time it took for a reboot to happen but the upgrade failed to progress.
- **flightID** Unique update ID
- **ForcedRebootReminderSet** A boolean value that indicates if a forced reboot will happen for updates.
- **rebootOutsideOfActiveHours** Indicates the timing that the reboot was to occur to ensure the correct update process and experience is provided to keep Windows up to date.
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date.
- **updateScenarioType** The update session type.

### **Microsoft.Windows.Update.Orchestrator.PreShutdownStart**

This event is generated right before the shutdown and commit operations

The following fields are available:

- **wuDeviceid** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue

### **Microsoft.Windows.Update.Orchestrator.DeferRestart**

This event indicates that a restart required for installing updates was postponed

The following fields are available:

- **filteredDeferReason** Indicates the raised, but ignorable, reasons that the USO didn't restart (for example, user active or low battery)
- **raisedDeferReason** Indicates the reason that the USO didn't restart. For example, user active or low battery
- **wuDeviceid** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue
- **eventScenario** Indicates the purpose of sending this event - whether because the software distribution just started checking for content, or whether it was cancelled, succeeded, or failed

### **Microsoft.Windows.Update.Orchestrator.DisplayNeeded**

Reboot postponed due to needing a display

The following fields are available:

- **displayNeededReason** Reason the display is needed
- **eventScenario** Indicates the purpose of sending this event - whether because the software distribution just started checking for content, or whether it was cancelled, succeeded, or failed
- **rebootOutsideOfActiveHours** Indicates the timing that the reboot was to occur to ensure the correct update process and experience is provided to keep Windows up to date
- **revisionNumber** Revision number of the update
- **updateId** Update ID
- **updateScenarioType** The update session type
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date
- **wuDeviceid** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue

### **Microsoft.Windows.Update.NotificationUx.RebootScheduled**

Indicates when a reboot is scheduled by the system or a user for a security, quality, or feature update

The following fields are available:

- **activeHoursApplicable** True, If Active Hours applicable on this device. False, otherwise
- **rebootArgument** Argument for the reboot task. It also represents specific reboot related action
- **rebootOutsideOfActiveHours** True, if a reboot is scheduled outside of active hours. False, otherwise
- **rebootScheduledByUser** True, if a reboot is scheduled by user. False, if a reboot is scheduled automatically
- **rebootState** The state of the reboot
- **revisionNumber** Revision number of the update that is getting installed with this reboot
- **scheduledRebootTime** Time of the scheduled reboot
- **updateId** ID of the update that is getting installed with this reboot
- **wuDeviceid** Unique device ID used by Windows Update
- **scheduledRebootTimeInUTC** Time of the scheduled reboot in Coordinated Universal Time



# Windows 10, version 1703 basic level Windows diagnostic events and fields

7/9/2018 • 158 minutes to read • [Edit Online](#)

## Applies to

- Windows 10, version 1703

The Basic level gathers a limited set of information that is critical for understanding the device and its configuration including: basic device information, quality-related information, app compatibility, and Microsoft Store. When the level is set to Basic, it also includes the Security level information. The Basic level also helps to identify problems that can occur on a particular device hardware or software configuration. For example, it can help determine if crashes are more frequent on devices with a specific amount of memory or that are running a particular driver version. This helps Microsoft fix operating system or app problems.

Use this article to learn about diagnostic events, grouped by event area, and the fields within each event. A brief description is provided for each field. Every event generated includes common data, which collects device data. You can learn more about Windows functional and diagnostic data through these articles:

- [Manage connections from Windows operating system components to Microsoft services](#)
- [Configure Windows diagnostic data in your organization](#)

### NOTE

Updated November 2017 to document new and modified events. We've added some new events and also added new fields to existing events to prepare for upgrades to the next release of Windows.

## Common data extensions

### Common Data Extensions.App

The following fields are available:

- **expld** Associates a flight, such as an OS flight, or an experiment, such as a web site UX experiment, with an event.
- **userid** The userID as known by the application.
- **env** The environment from which the event was logged.
- **asld** An integer value that represents the app session. This value starts at 0 on the first app launch and increments after each subsequent app launch per boot session.

### Common Data Extensions.CS

The following fields are available:

- **sig** A common schema signature that identifies new and modified event schemas.

### Common Data Extensions.CUET

The following fields are available:

- **stld** Represents the Scenario Entry Point ID. This is a unique GUID for each event in a diagnostic scenario. This used to be Scenario Trigger ID.
- **ald** Represents the ETW ActivityId. Logged via TraceLogging or directly via ETW.

- **rald** Represents the ETW Related ActivityId. Logged via TraceLogging or directly via ETW.
- **op** Represents the ETW Op Code.
- **cat** Represents a bitmask of the ETW Keywords associated with the event.
- **flags** Represents the bitmap that captures various Windows specific flags.
- **cpId** The composer ID, such as Reference, Desktop, Phone, Holographic, Hub, IoT Composer.
- **tickets** A list of strings that represent entries in the HTTP header of the web request that includes this event.
- **bseq** Upload buffer sequence number in the format <buffer identifier>:<sequence number>
- **mon** Combined monitor and event sequence numbers in the format <monitor sequence>:<event sequence>

#### Common Data Extensions.Device

The following fields are available:

- **ver** Represents the major and minor version of the extension.
- **localId** Represents a locally defined unique ID for the device, not the human readable device name. Most likely equal to the value stored at HKLM\Software\Microsoft\SQMClient\MachineId
- **deviceClass** Represents the classification of the device, the device "family". For example, Desktop, Server, or Mobile.

#### Common Data Extensions.Envelope

The following fields are available:

- **ver** Represents the major and minor version of the extension.
- **name** Represents the uniquely qualified name for the event.
- **time** Represents the event date time in Coordinated Universal Time (UTC) when the event was generated on the client. This should be in ISO 8601 format.
- **popSample** Represents the effective sample rate for this event at the time it was generated by a client.
- **epoch** Represents the epoch and seqNum fields, which help track how many events were fired and how many events were uploaded, and enables identification of data lost during upload and de-duplication of events on the ingress server.
- **seqNum** Represents the sequence field used to track absolute order of uploaded events. It is an incrementing identifier for each event added to the upload queue. The Sequence helps track how many events were fired and how many events were uploaded and enables identification of data lost during upload and de-duplication of events on the ingress server.
- **iKey** Represents an ID for applications or other logical groupings of events.
- **flags** Represents a collection of bits that describe how the event should be processed by the Connected User Experiences and Telemetry component pipeline. The lowest-order byte is the event persistence. The next byte is the event latency.
- **os** Represents the operating system name.
- **osVer** Represents the OS version, and its format is OS dependent.
- **appId** Represents a unique identifier of the client application currently loaded in the process producing the event; and is used to group events together and understand usage pattern, errors by application.
- **appVer** Represents the version number of the application. Used to understand errors by Version, Usage by Version across an app.
- **cV** Represents the Correlation Vector: A single field for tracking partial order of related diagnostic data events across component boundaries.

#### Common Data Extensions.OS

The following fields are available:

- **ver** Represents the major and minor version of the extension.
- **expId** Represents the experiment ID. The standard for associating a flight, such as an OS flight (pre-release

build), or an experiment, such as a web site UX experiment, with an event is to record the flight / experiment IDs in Part A of the common schema.

- **locale** Represents the locale of the operating system.
- **bootId** An integer value that represents the boot session. This value starts at 0 on first boot after OS install and increments after every reboot.

#### Common Data Extensions.User

The following fields are available:

- **ver** Represents the major and minor version of the extension.
- **localId** Represents a unique user identity that is created locally and added by the client. This is not the user's account ID.

#### Common Data Extensions.XBL

The following fields are available:

- **nbf** Not before time
- **expId** Expiration time
- **sbx** XBOX sandbox identifier
- **dtv** XBOX device type
- **did** XBOX device ID
- **xid** A list of base10-encoded XBOX User IDs.
- **uts** A bit field, with 2 bits being assigned to each user ID listed in xid. This field is omitted if all users are retail accounts.

#### Common Data Extensions.Consent UI Event

This User Account Control (UAC) diagnostic data point collects information on elevations that originate from low integrity levels. This occurs when a process running at low integrity level (IL) requires higher (administrator) privileges, and therefore requests for elevation via UAC (consent.exe). By better understanding the processes requesting these elevations, Microsoft can in turn improve the detection and handling of potentially malicious behavior in this path.

The following fields are available:

- **eventType** Represents the type of elevation: If it succeeded, was cancelled, or was auto-approved.
- **splitToken** Represents the flag used to distinguish between administrators and standard users.
- **friendlyName** Represents the name of the file requesting elevation from low IL.
- **elevationReason** Represents the distinction between various elevation requests sources (appcompat, installer, COM, MSI and so on).
- **exeName** Represents the name of the file requesting elevation from low IL.
- **signatureState** Represents the state of the signature, if it signed, unsigned, OS signed and so on.
- **publisherName** Represents the name of the publisher of the file requesting elevation from low IL.
- **cmdLine** Represents the full command line arguments being used to elevate.
- **Hash.Length** Represents the length of the hash of the file requesting elevation from low IL.
- **Hash** Represents the hash of the file requesting elevation from low IL.
- **HashAlgId** Represents the algorithm ID of the hash of the file requesting elevation from low IL.
- **telemetryFlags** Represents the details about the elevation prompt for CEIP data.
- **timeStamp** Represents the time stamp on the file requesting elevation.
- **fileVersionMS** Represents the major version of the file requesting elevation.
- **fileVersionLS** Represents the minor version of the file requesting elevation.

# Common data fields

## Common Data Fields.Ms.Device.DeviceInventory.Change

These fields are added whenever Ms.Device.DeviceInventoryChange is included in the event.

The following fields are available:

- **syncId** A string used to group StartSync, EndSync, Add, and Remove operations that belong together. This field is unique by Sync period and is used to disambiguate in situations where multiple agents perform overlapping inventories for the same object.
- **objectType** Indicates the object type that the event applies to.
- **Action** The change that was invoked on a device inventory object.
- **inventoryId** Device ID used for Compatibility testing

## Common Data Fields.TelClientSynthetic.PrivacySettingsAfterCreatorsUpdate.PreUpgradeSettings

These fields are added whenever PreUpgradeSettings is included in the event.

The following fields are available:

- **HKLM\_SensorPermissionState.SensorPermissionState** The state of the Location service before the feature update completed.
- **HKLM\_SensorPermissionState.HRESULT** The error code returned when trying to query the Location service for the device.
- **HKCU\_SensorPermissionState.SensorPermissionState** The state of the Location service when a user signs on before the feature update completed.
- **HKCU\_SensorPermissionState.HRESULT** The error code returned when trying to query the Location service for the current user.
- **HKLM\_LocationPlatform.Status** The state of the location platform after the feature update has completed.
- **HKLM\_LocationPlatform.HRESULT** The error code returned when trying to query the location platform for the device.
- **HKLM\_LocationSyncEnabled.AcceptedPrivacyPolicy** The speech recognition state for the device before the feature update completed.
- **HKLM\_LocationSyncEnabled.HRESULT** The error code returned when trying to query the Find My Device service for the device.
- **HKCU\_LocationSyncEnabled.AcceptedPrivacyPolicy** The speech recognition state for the current user before the feature update completed.
- **HKCU\_LocationSyncEnabled.HRESULT** The error code returned when trying to query the Find My Device service for the current user.
- **HKLM\_AllowTelemetry.AllowTelemetry** The state of the Connected User Experiences and Telemetry component for the device before the feature update.
- **HKLM\_AllowTelemetry.HRESULT** The error code returned when trying to query the Connected User Experiences and Telemetry component for the device.
- **HKLM\_TIPC.Enabled** The state of TIPC for the device.
- **HKLM\_TIPC.HRESULT** The error code returned when trying to query TIPC for the device.
- **HKCU\_TIPC.Enabled** The state of TIPC for the current user.
- **HKCU\_TIPC.HRESULT** The error code returned when trying to query TIPC for the current user.
- **HKLM\_FlipAhead.FPEnabled** Is Flip Ahead enabled for the device before the feature update was completed?
- **HKLM\_FlipAhead.HRESULT** The error code returned when trying to query Flip Ahead for the device.
- **HKCU\_FlipAhead.FPEnabled** Is Flip Ahead enabled for the current user before the feature update was completed?
- **HKCU\_FlipAhead.HRESULT** The error code returned when trying to query Flip Ahead for the current user.

- **HKLM\_TailoredExperiences.TailoredExperiencesWithDiagnosticDataEnabled** Is Tailored Experiences with Diagnostics Data enabled for the current user after the feature update had completed?
- **HKCU\_TailoredExperiences.HRESULT** The error code returned when trying to query Tailored Experiences with Diagnostics Data for the current user.
- **HKLM\_AdvertisingID.Enabled** Is the advertising ID enabled for the device?
- **HKLM\_AdvertisingID.HRESULT** The error code returned when trying to query the state of the advertising ID for the device.
- **HKCU\_AdvertisingID.Enabled** Is the advertising ID enabled for the current user?
- **HKCU\_AdvertisingID.HRESULT** The error code returned when trying to query the state of the advertising ID for the user.

#### **Common Data Fields.TelClientSynthetic.PrivacySettingsAfterCreatorsUpdate.PostUpgradeSettings**

These fields are added whenever PostUpgradeSettings is included in the event.

The following fields are available:

- **HKLM\_SensorPermissionState.SensorPermissionState** The state of the Location service after the feature update has completed.
- **HKLM\_SensorPermissionState.HRESULT** The error code returned when trying to query the Location service for the device.
- **HKCU\_SensorPermissionState.SensorPermissionState** The state of the Location service when a user signs on after a feature update has completed.
- **HKCU\_SensorPermissionState.HRESULT** The error code returned when trying to query the Location service for the current user.
- **HKLM\_LocationPlatform.Status** The state of the location platform after the feature update has completed.
- **HKLM\_LocationPlatform.HRESULT** The error code returned when trying to query the location platform for the device.
- **HKLM\_LocationSyncEnabled.AcceptedPrivacyPolicy** The speech recognition state for the device after the feature update has completed.
- **HKLM\_LocationSyncEnabled.HRESULT** The error code returned when trying to query the Find My Device service for the device.
- **HKCU\_LocationSyncEnabled.AcceptedPrivacyPolicy** The speech recognition state for the current user after the feature update has completed.
- **HKCU\_LocationSyncEnabled.HRESULT** The error code returned when trying to query the Find My Device service for the current user.
- **HKLM\_AllowTelemetry.AllowTelemetry** The state of the Connected User Experiences and Telemetry component for the device after the feature update.
- **HKLM\_AllowTelemetry.HRESULT** The error code returned when trying to query the Connected User Experiences and Telemetry component for the device.
- **HKLM\_TIPC.Enabled** The state of TIPC for the device.
- **HKLM\_TIPC.HRESULT** The error code returned when trying to query TIPC for the device.
- **HKCU\_TIPC.Enabled** The state of TIPC for the current user.
- **HKCU\_TIPC.HRESULT** The error code returned when trying to query TIPC for the current user.
- **HKLM\_FlipAhead.FPEnabled** Is Flip Ahead enabled for the device after the feature update has completed?
- **HKLM\_FlipAhead.HRESULT** The error code returned when trying to query Flip Ahead for the device.
- **HKCU\_FlipAhead.FPEnabled** Is Flip Ahead enabled for the current user after the feature update has completed?
- **HKCU\_FlipAhead.HRESULT** The error code returned when trying to query Flip Ahead for the current user.
- **HKLM\_TailoredExperiences.TailoredExperiencesWithDiagnosticDataEnabled** Is Tailored Experiences with Diagnostics Data enabled for the current user after the feature update had completed?

- **HKCU\_TailoredExperiences.HRESULT** The error code returned when trying to query Tailored Experiences with Diagnostics Data for the current user.
- **HKLM\_AdvertisingID.Enabled** Is the advertising ID enabled for the device?
- **HKLM\_AdvertisingID.HRESULT** The error code returned when trying to query the state of the advertising ID for the device.
- **HKCU\_AdvertisingID.Enabled** Is the advertising ID enabled for the current user?
- **HKCU\_AdvertisingID.HRESULT** The error code returned when trying to query the state of the advertising ID for the user.

## Appraiser events

### Microsoft.Windows.Appraiser.General.ChecksumTotalPictureCount

This event lists the types of objects and how many of each exist on the client device. This allows for a quick way to ensure that the records present on the server match what is present on the client.

The following fields are available:

- **DatasourceApplicationFile\_RS3** The total DecisionApplicationFile objects targeting the next release of Windows on this device.
- **DatasourceDevicePnp\_RS3** The total DatasourceDevicePnp objects targeting the next release of Windows on this device.
- **DatasourceDriverPackage\_RS3** The total DatasourceDriverPackage objects targeting the next release of Windows on this device.
- **DataSourceMatchingInfoBlock\_RS3** The total DataSourceMatchingInfoBlock objects targeting the next release of Windows on this device.
- **DataSourceMatchingInfoPassive\_RS3** The total DataSourceMatchingInfoPassive objects targeting the next release of Windows on this device.
- **DataSourceMatchingInfoPostUpgrade\_RS3** The total DataSourceMatchingInfoPostUpgrade objects targeting the next release of Windows on this device.
- **DatasourceSystemBios\_RS3** The total DatasourceSystemBios objects targeting the next release of Windows on this device.
- **DecisionApplicationFile\_RS3** The total DecisionApplicationFile objects targeting the next release of Windows on this device.
- **DecisionDevicePnp\_RS3** The total DecisionDevicePnp objects targeting the next release of Windows on this device.
- **DecisionDriverPackage\_RS3** The total DecisionDriverPackage objects targeting the next release of Windows on this device.
- **DecisionMatchingInfoBlock\_RS3** The total DecisionMatchingInfoBlock objects targeting the next release of Windows on this device.
- **DecisionMatchingInfoPassive\_RS3** The total DataSourceMatchingInfoPostUpgrade objects targeting the next release of Windows on this device.
- **DecisionMatchingInfoPostUpgrade\_RS3** The total DecisionMatchingInfoPostUpgrade objects targeting the next release of Windows on this device.
- **DecisionMediaCenter\_RS3** The total DecisionMediaCenter objects targeting the next release of Windows on this device.
- **DecisionSystemBios\_RS3** The total DecisionSystemBios objects targeting the next release of Windows on this device.
- **PCFP** An ID for the system that is calculated by hashing hardware identifiers.
- **InventoryApplicationFile** The total InventoryApplicationFile objects that are present on this device.
- **InventoryMediaCenter** The total InventoryMediaCenter objects that are present on this device.

- **InventoryLanguagePack** The total InventoryLanguagePack objects that are present on this device.
- **InventoryUplevelDriverPackage** The total InventoryUplevelDriverPackage objects that are present on this device.
- **InventorySystemBios** The total InventorySystemBios objects that are present on this device.
- **SystemProcessorCompareExchange** The total SystemProcessorCompareExchange objects that are present on this device.
- **SystemProcessorLahfSahf** The total SystemProcessorLahfSahf objects that are present on this device.
- **SystemMemory** The total SystemMemory objects that are present on this device.
- **SystemProcessorPrefetchW** The total SystemProcessorPrefetchW objects that are present on this device.
- **SystemProcessorSse2** The total SystemProcessorSse2 objects that are present on this device.
- **SystemProcessorNx** The total SystemProcessorNx objects that are present on this device.
- **SystemWlan** The total SystemWlan objects that are present on this device.
- **SystemWim** The total SystemWim objects that are present on this device.
- **SystemTouch** The total SystemTouch objects that are present on this device.
- **SystemWindowsActivationStatus** The total SystemWindowsActivationStatus objects that are present on this device.
- **Wmdrm\_RS3** The total Wmdrm objects targeting the next release of Windows on this device.

#### **Microsoft.Windows.Appraiser.General.ChecksumTotalPictureIdHashSha256**

This event lists the types of objects and the hashed values of all the identifiers for each one. This allows for a more in-depth way to ensure that the records present on the server match what is present on the client.

The following fields are available:

- **DatasourceApplicationFile\_RS3** The total DatasourceApplicationFile objects targeting the next release of Windows on this device.
- **DatasourceDevicePnp\_RS3** The total DatasourceDevicePnp objects targeting the next release of Windows on this device.
- **DatasourceDriverPackage\_RS3** The total DatasourceDriverPackage objects targeting the next release of Windows on this device.
- **DataSourceMatchingInfoBlock\_RS3** The total DataSourceMatchingInfoBlock objects targeting the next release of Windows on this device.
- **DataSourceMatchingInfoPassive\_RS3** The total DataSourceMatchingInfoPassive objects targeting the next release of Windows on this device.
- **DataSourceMatchingInfoPostUpgrade\_RS3** The total DataSourceMatchingInfoPostUpgrade objects targeting the next release of Windows on this device.
- **DatasourceSystemBios\_RS3** The total DatasourceSystemBios objects targeting the next release of Windows on this device.
- **DecisionApplicationFile\_RS3** The total DecisionApplicationFile objects targeting the next release of Windows on this device.
- **DecisionDevicePnp\_RS3** The total DecisionDevicePnp objects targeting the next release of Windows on this device.
- **DecisionDriverPackage\_RS3** The total DecisionDriverPackage objects targeting the next release of Windows on this device.
- **DecisionMatchingInfoBlock\_RS3** The total DecisionMatchingInfoBlock objects targeting the next release of Windows on this device.
- **DecisionMatchingInfoPassive\_RS3** The total DataSourceMatchingInfoPostUpgrade objects targeting the next release of Windows on this device.
- **DecisionMatchingInfoPostUpgrade\_RS3** The total DecisionMatchingInfoPostUpgrade objects targeting the next release of Windows on this device.

- **DecisionMediaCenter\_RS3** The total DecisionMediaCenter objects targeting the next release of Windows on this device.
- **DecisionSystemBios\_RS3** The total DecisionSystemBios objects targeting the next release of Windows on this device.
- **PCFP** An ID for the system that is calculated by hashing hardware identifiers.
- **InventoryApplicationFile** The SHA256 hash of InventoryApplicationFile objects that are present on this device.
- **InventoryMediaCenter** The SHA256 hash of InventoryMediaCenter objects that are present on this device.
- **InventoryLanguagePack** The SHA256 hash of InventoryLanguagePack objects that are present on this device.
- **InventoryUplevelDriverPackage** The SHA256 hash of InventoryUplevelDriverPackage objects that are present on this device.
- **InventorySystemBios** The SHA256 hash of InventorySystemBios objects that are present on this device.
- **SystemProcessorCompareExchange** The SHA256 hash of SystemProcessorCompareExchange objects that are present on this device.
- **SystemProcessorLahfSahf** The SHA256 hash of SystemProcessorLahfSahf objects that are present on this device.
- **SystemMemory** The SHA256 hash of SystemMemory objects that are present on this device.
- **SystemProcessorPrefetchW** The SHA256 hash of SystemProcessorPrefetchW objects that are present on this device.
- **SystemProcessorSse2** The SHA256 hash of SystemProcessorSse2 objects that are present on this device.
- **SystemProcessorNx** The SHA256 hash of SystemProcessorNx objects that are present on this device.
- **SystemWlan** The SHA256 hash of SystemWlan objects that are present on this device.
- **SystemWim** The SHA256 hash of SystemWim objects that are present on this device.
- **SystemTouch** The SHA256 hash of SystemTouch objects that are present on this device.
- **SystemWindowsActivationStatus** The SHA256 hash of SystemWindowsActivationStatus objects that are present on this device.
- **Wmdrm\_RS3** The total Wmdrm objects targeting the next release of Windows on this device.

#### **Microsoft.Windows.Appraiser.General.DatasourceApplicationFileAdd**

This event sends compatibility information about a file to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file that is generating the events.
- **AvDisplayName** If it is an anti-virus app, this is its display name.
- **CompatModelIndex** The compatibility prediction for this file.
- **HasCitData** Is the file present in CIT data?
- **HasUpgradeExe** Does the anti-virus app have an upgrade.exe file?
- **IsAv** Is the file an anti-virus reporting EXE?
- **ResolveAttempted** This will always be an empty string when sending diagnostic data.
- **SdbEntries** An array of fields that indicates the SDB entries that apply to this file.

#### **Microsoft.Windows.Appraiser.General.DatasourceApplicationFileRemove**

This event indicates that the DatasourceApplicationFile object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DatasourceApplicationFileStartSync**



This event indicates that a new set of DatasourceApplicationFileAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DatasourceDevicePnpAdd**

This event sends compatibility data for a PNP device, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **ActiveNetworkConnection** Is the device an active network device?
- **IsBootCritical** Is the device boot critical?
- **SdbEntries** An array of fields indicating the SDB entries that apply to this device.
- **WuDriverCoverage** Is there a driver uplevel for this device according to Windows Update?
- **WuDriverUpdateID** The Windows Update ID of the applicable uplevel driver.
- **WuPopulatedFromID** The expected uplevel driver matching ID based on driver coverage from Windows Update.

#### **Microsoft.Windows.Appraiser.General.DatasourceDevicePnpRemove**

This event indicates that the DatasourceDevicePnp object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DatasourceDevicePnpStartSync**

This event indicates that a new set of DatasourceDevicePnpAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DatasourceDriverPackageAdd**

This event sends compatibility database data about driver packages to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **SdbEntries** An array of fields indicating the SDB entries that apply to this driver package.

#### **Microsoft.Windows.Appraiser.General.DatasourceDriverPackageRemove**

This event indicates that the DatasourceDriverPackage object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DatasourceDriverPackageStartSync**

This event indicates that a new set of DatasourceDriverPackageAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoBlockAdd**

This event sends blocking data about any compatibility blocking entries hit on the system that are not directly related to specific applications or devices, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **SdbEntries** An array of fields indicating the SDB entries that apply to this file.

#### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoBlockRemove**

This event indicates that the DataSourceMatchingInfoBlock object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoBlockStartSync**

This event indicates that a full set of DataSourceMatchingInfoBlockStAdd events have been sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPassiveAdd**

This event sends compatibility database information about non-blocking compatibility entries on the system that are not keyed by either applications or devices, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **SdbEntries** An array of fields indicating the SDB entries that apply to this file.

#### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPassiveRemove**

This event indicates that the DataSourceMatchingInfoPassive object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPassiveStartSync**

This event indicates that a new set of DataSourceMatchingInfoPassiveAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPostUpgradeAdd**

This event sends compatibility database information about entries requiring reinstallation after an upgrade on the system that are not keyed by either applications or devices, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **SdbEntries** An array of fields indicating the SDB entries that apply to this file.

#### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPostUpgradeRemove**

This event indicates that the DataSourceMatchingInfoPostUpgrade object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPostUpgradeStartSync**

This event indicates that a new set of DataSourceMatchingInfoPostUpgradeAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DatasourceSystemBiosAdd**

This event sends compatibility database information about the BIOS to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **SdbEntries** An array of fields indicating the SDB entries that apply to this BIOS.

#### **Microsoft.Windows.Appraiser.General.DatasourceSystemBiosRemove**

This event indicates that the DatasourceSystemBios object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DatasourceSystemBiosStartSync**

This event indicates that a new set of DatasourceSystemBiosAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionApplicationFileAdd**

This event sends compatibility decision data about a file to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **BlockAlreadyInbox** The uplevel runtime block on the file already existed on the current OS.
- **BlockingApplication** Are there any application issues that interfere with upgrade due to the file in question?
- **DisplayGenericMessage** Will be a generic message be shown for this file?
- **HardBlock** This file is blocked in the SDB.
- **HasUxBlockOverride** Does the file have a block that is overridden by a tag in the SDB?
- **MigApplication** Does the file have a MigXML from the SDB associated with it that applies to the current upgrade mode?
- **MigRemoval** Does the file have a MigXML from the SDB that will cause the app to be removed on upgrade?
- **NeedsDismissAction** Will the file cause an action that can be dismissed?
- **NeedsInstallPostUpgradeData** After upgrade, the file will have a post-upgrade notification to install a replacement for the app.
- **NeedsNotifyPostUpgradeData** Does the file have a notification that should be shown after upgrade?
- **NeedsReinstallPostUpgradeData** After upgrade, this file will have a post-upgrade notification to reinstall the app.
- **NeedsUninstallAction** The file must be uninstalled to complete the upgrade.
- **SdbBlockUpgrade** The file is tagged as blocking upgrade in the SDB,
- **SdbBlockUpgradeCanReinstall** The file is tagged as blocking upgrade in the SDB. It can be reinstalled after upgrade.
- **SdbBlockUpgradeUntilUpdate** The file is tagged as blocking upgrade in the SDB. If the app is updated, the upgrade can proceed.
- **SdbReinstallUpgrade** The file is tagged as needing to be reinstalled after upgrade in the SDB. It does not block upgrade.
- **SdbReinstallUpgradeWarn** The file is tagged as needing to be reinstalled after upgrade with a warning in the

SDB. It does not block upgrade.

- **SoftBlock** The file is softblocked in the SDB and has a warning.

#### **Microsoft.Windows.Appraiser.General.DecisionApplicationFileRemove**

This event indicates that the DecisionApplicationFile object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionApplicationFileStartSync**

This event indicates that a new set of DecisionApplicationFileAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionDevicePnpAdd**

This event sends compatibility decision data about a PNP device to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **AssociatedDriverIsBlocked** Is the driver associated with this PNP device blocked?
- **BlockAssociatedDriver** Should the driver associated with this PNP device be blocked?
- **BlockUpgradeIfDriverBlocked** Is the PNP device both boot critical and does not have a driver included with the OS?
- **BlockUpgradeIfDriverBlockedAndOnlyActiveNetwork** Is this PNP device the only active network device?
- **BlockingDevice** Is this PNP device blocking upgrade?
- **DisplayGenericMessage** Will a generic message be shown during Setup for this PNP device?
- **DriverAvailableInbox** Is a driver included with the operating system for this PNP device?
- **DriverAvailableOnline** Is there a driver for this PNP device on Windows Update?
- **DriverAvailableUplevel** Is there a driver on Windows Update or included with the operating system for this PNP device?
- **DriverBlockOverridden** Is there is a driver block on the device that has been overridden?
- **NeedsDismissAction** Will the user would need to dismiss a warning during Setup for this device?
- **NotRegressed** Does the device have a problem code on the source OS that is no better than the one it would have on the target OS?
- **SdbDeviceBlockUpgrade** Is there an SDB block on the PNP device that blocks upgrade?
- **SdbDriverBlockOverridden** Is there an SDB block on the PNP device that blocks upgrade, but that block was overridden?
- **AssociatedDriverWillNotMigrate** Will the driver associated with this plug-and-play device migrate?

#### **Microsoft.Windows.Appraiser.General.DecisionDevicePnpRemove**

This event indicates that the DecisionDevicePnp object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionDevicePnpStartSync**

This event indicates that the DecisionDevicePnp object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.DecisionDriverPackageAdd**

This event sends decision data about driver package compatibility to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **DriverBlockOverridden** Does the driver package have an SDB block that blocks it from migrating, but that block has been overridden?
- **DriverIsDeviceBlocked** Was the driver package was blocked because of a device block?
- **DriverIsDriverBlocked** Is the driver package blocked because of a driver block?
- **DriverShouldNotMigrate** Should the driver package be migrated during upgrade?
- **SdbDriverBlockOverridden** Does the driver package have an SDB block that blocks it from migrating, but that block has been overridden?

### **Microsoft.Windows.Appraiser.General.DecisionDriverPackageRemove**

This event indicates that the DecisionDriverPackage object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.DecisionDriverPackageStartSync**

This event indicates that a new set of DecisionDriverPackageAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoBlockAdd**

This event sends compatibility decision data about blocking entries on the system that are not keyed by either applications or devices, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **BlockingApplication** Are there are any application issues that interfere with upgrade due to matching info blocks?
- **DisplayGenericMessage** Will a generic message be shown for this block?
- **NeedsUninstallAction** Does the user need to take an action in setup due to a matching info block?
- **SdbBlockUpgrade** Is a matching info block blocking upgrade?
- **SdbBlockUpgradeCanReinstall** Is a matching info block blocking upgrade, but has the can reinstall tag?
- **SdbBlockUpgradeUntilUpdate** Is a matching info block blocking upgrade but has the until update tag?

### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoBlockRemove**

This event indicates that the DecisionMatchingInfoBlock object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoBlockStartSync**

This event indicates that a new set of DecisionMatchingInfoBlockAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPassiveAdd**

This event sends compatibility decision data about non-blocking entries on the system that are not keyed by either applications or devices, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **BlockingApplication** Are there any application issues that interfere with upgrade due to matching info blocks?
- **MigApplication** Is there a matching info block with a mig for the current mode of upgrade?

### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPassiveRemove**

This event Indicates that the DecisionMatchingInfoPassive object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPassiveStartSync**

This event indicates that a new set of DecisionMatchingInfoPassiveAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPostUpgradeAdd**

This event sends compatibility decision data about entries that require reinstall after upgrade. It's used to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **NeedsInstallPostUpgradeData** Will the file have a notification after upgrade to install a replacement for the app?
- **NeedsNotifyPostUpgradeData** Should a notification be shown for this file after upgrade?
- **NeedsReinstallPostUpgradeData** Will the file have a notification after upgrade to reinstall the app?
- **SdbReinstallUpgrade** The file is tagged as needing to be reinstalled after upgrade in the compatibility database (but is not blocking upgrade).

### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPostUpgradeRemove**

This event indicates that the DecisionMatchingInfoPostUpgrade object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPostUpgradeStartSync**

This event indicates that a new set of DecisionMatchingInfoPostUpgradeAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

### **Microsoft.Windows.Appraiser.General.DecisionMediaCenterAdd**

This event sends decision data about the presence of Windows Media Center, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.

- **BlockingApplication** Is there any application issues that interfere with upgrade due to Windows Media Center?
- **MediaCenterActivelyUsed** If Windows Media Center is supported on the edition, has it been run at least once and are the MediaCenterIndicators are true?
- **MediaCenterInUse** Is Windows Media Center actively being used?
- **MediaCenterIndicators** Do any indicators imply that Windows Media Center is in active use?
- **MediaCenterPaidOrActivelyUsed** Is Windows Media Center actively being used or is it running on a supported edition?
- **NeedsDismissAction** Are there any actions that can be dismissed coming from Windows Media Center?

#### **Microsoft.Windows.Appraiser.General.DecisionMediaCenterRemove**

This event indicates that the DecisionMediaCenter object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionMediaCenterStartSync**

This event indicates that a new set of DecisionMediaCenterAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionSystemBiosAdd**

This event sends compatibility decision data about the BIOS to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **Blocking** Is the device blocked from upgrade due to a BIOS block?
- **HasBiosBlock** Does the device have a BIOS block?

#### **Microsoft.Windows.Appraiser.General.DecisionSystemBiosRemove**

This event indicates that the DecisionSystemBios object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.DecisionSystemBiosStartSync**

This event indicates that a new set of DecisionSystemBiosAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.EnterpriseScenarioWithDiagTrackServiceRunning**

The event that indicates that Appraiser has been triggered to run an enterprise scenario while the DiagTrack service is installed. This event can only be sent if a special flag is used to trigger the enterprise scenario.

The following fields are available:

- **Time** The client time of the event.
- **PCFP** An ID for the system calculated by hashing hardware identifiers.

#### **Microsoft.Windows.Appraiser.General.GatedRegChange**

This event sends data about the results of running a set of quick-blocking instructions, to help keep Windows up to

date.

The following fields are available:

- **Time** The client time of the event.
- **PCFP** An ID for the system calculated by hashing hardware identifiers.
- **RegKey** The registry key name for which a result is being sent.
- **RegValue** The registry value for which a result is being sent.
- **OldData** The previous data in the registry value before the scan ran.
- **NewData** The data in the registry value after the scan completed.

#### **Microsoft.Windows.Appraiser.General.InventoryApplicationFileAdd**

This event represents the basic metadata about a file on the system. The file must be part of an app and either have a block in the compatibility database or are part of an anti-virus program.

The following fields are available:

- **AvDisplayName** If the app is an anti-virus app, this is its display name.
- **AvProductState** Represents state of antivirus program with respect to whether it's turned on and the signatures are up-to-date.
- **BinaryType** A binary type. Example: UNINITIALIZED, ZERO\_BYTE, DATA\_ONLY, DOS\_MODULE, NE16\_MODULE, PE32\_UNKNOWN, PE32\_I386, PE32\_ARM, PE64\_UNKNOWN, PE64\_AMD64, PE64\_ARM64, PE64\_IA64, PE32\_CLR\_32, PE32\_CLR\_IL, PE32\_CLR\_IL\_PREFER32, PE64\_CLR\_64
- **BinFileVersion** An attempt to clean up FileVersion at the client that tries to place the version into 4 octets.
- **BinProductVersion** An attempt to clean up ProductVersion at the client that tries to place the version into 4 octets.
- **BoeProgramId** If there is no entry in Add/Remove Programs, this is the ProgramID that is generated from the file metadata.
- **CompanyName** The company name of the vendor who developed this file.
- **FileId** A hash that uniquely identifies a file.
- **FileVersion** The File version field from the file metadata under Properties -> Details.
- **HasUpgradeExe** Does the anti-virus app have an upgrade.exe file?
- **IsAv** Is the file an anti-virus reporting EXE?
- **LinkDate** The date and time that this file was linked on.
- **LowerCaseLongPath** The full file path to the file that was inventoried on the device.
- **Name** The name of the file that was inventoried.
- **ProductName** The Product name field from the file metadata under Properties -> Details.
- **ProductVersion** The Product version field from the file metadata under Properties -> Details.
- **ProgramId** A hash of the Name, Version, Publisher, and Language of an application used to identify it.
- **Size** The size of the file (in hexadecimal bytes).

#### **Microsoft.Windows.Inventory.Core.InventoryApplicationDriverAdd**

This event represents the drivers that an application installs.

The following fields are available:

- **InventoryVersion** The version of the inventory component
- **Programids** The unique program identifier the driver is associated with.

#### **Microsoft.Windows.Inventory.Core.InventoryApplicationDriverStartSync**

This event indicates that a new set of InventoryApplicationDriverStartAdd events will be sent.

The following fields are available:



- **InventoryVersion** The version of the inventory component.

#### **Microsoft.Windows.Appraiser.General.InventoryApplicationFileRemove**

This event indicates that the InventoryApplicationFile object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.InventoryApplicationFileStartSync**

This event indicates that a new set of InventoryApplicationFileAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.InventoryLanguagePackAdd**

This event sends data about the number of language packs installed on the system, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **HasLanguagePack** Does this device have 2 or more language packs?
- **LanguagePackCount** How many language packs are installed?

#### **Microsoft.Windows.Appraiser.General.InventoryLanguagePackRemove**

This event indicates that the InventoryLanguagePack object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.InventoryLanguagePackStartSync**

This event indicates that a new set of InventoryLanguagePackAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.InventoryMediaCenterAdd**

This event sends true/false data about decision points used to understand whether Windows Media Center is used on the system, to help keep Windows up to date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **EverLaunched** Has Windows Media Center ever been launched?
- **HasConfiguredTv** Has the user configured a TV tuner through Windows Media Center?
- **HasExtendedUserAccounts** Are any Windows Media Center Extender user accounts configured?
- **HasWatchedFolders** Are any folders configured for Windows Media Center to watch?
- **IsDefaultLauncher** Is Windows Media Center the default app for opening music or video files?
- **IsPaid** Is the user running a Windows Media Center edition that implies they paid for Windows Media Center?
- **IsSupported** Does the running OS support Windows Media Center?

#### **Microsoft.Windows.Appraiser.General.InventoryMediaCenterRemove**

This event indicates that the InventoryMediaCenter object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.InventoryMediaCenterStartSync**

This event indicates that a new set of InventoryMediaCenterAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.InventorySystemBiosAdd**

This event sends basic metadata about the BIOS to determine whether it has a compatibility block.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **BiosDate** The release date of the BIOS in UTC format.
- **BiosName** The name field from Win32\_BIOS.
- **Manufacturer** The manufacturer field from Win32\_ComputerSystem.
- **Model** The model field from Win32\_ComputerSystem.

#### **Microsoft.Windows.Appraiser.General.InventorySystemBiosRemove**

This event indicates that the InventorySystemBios object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.InventorySystemBiosStartSync**

This event indicates that a new set of InventorySystemBiosAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.InventoryUplevelDriverPackageAdd**

This event is only runs during setup. It provides a listing of the uplevel driver packages that were downloaded before the upgrade. Is critical to understanding if failures in setup can be traced to not having sufficient uplevel drivers before the upgrade.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **BootCritical** Is the driver package marked as boot critical?
- **Build** The build value from the driver package.
- **CatalogFile** The name of the catalog file within the driver package.
- **ClassGuid** The device class GUID from the driver package.
- **Class** The device class from the driver package.
- **Date** The date from the driver package.
- **SignatureStatus** Indicates if the driver package is signed. Unknown:0, Unsigned:1, Signed: 2
- **Inbox** Is the driver package of a driver that is included with Windows?
- **VersionMajor** The major version of the driver package.
- **VersionMinor** The minor version of the driver package.
- **OriginalName** The original name of the INF file before it was renamed. Generally a path under \$WINDOWS.~BT\Drivers\DU

- **Provider** The provider of the driver package.
- **PublishedName** The name of the INF file, post-rename.
- **Revision** The revision of the driver package.

#### **Microsoft.Windows.Appraiser.General.InventoryUplevelDriverPackageRemove**

This event indicates that the InventoryUplevelDriverPackage object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.InventoryUplevelDriverPackageStartSync**

This event indicates that a new set of InventoryUplevelDriverPackageAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.IsOnlineTelemetryOutputter**

This event indicates if Appraiser was able to connect successfully to Windows Update to get driver availability information.

The following fields are available:

- **Time** The client time of the event.
- **PCFP** A unique hardware identifier that is calculated by hashing hardware identifiers.
- **IsOnlineRun** Was the device able to connect to Windows Update to get driver availability information?

#### **Microsoft.Windows.Appraiser.General.IsOnlineWuDriverDataSource**

This event indicates if Appraiser was able to connect to Windows Update to gather driver coverage information.

The following fields are available:

- **Time** The client time of the event.
- **PCFP** A unique hardware identifier that is calculated by hashing hardware identifiers.
- **IsOnlineRun** Was the device able to connect to Windows Update to get driver availability information?
- **TargetVersion** The abbreviated name for the OS version against which Windows Update was queried.

#### **Microsoft.Windows.Appraiser.General.RunContext**

This event indicates what should be expected in the data payload.

The following fields are available:

- **AppraiserBranch** The source branch in which the currently running version of Appraiser was built.
- **AppraiserVersion** The version of the Appraiser file generating the events.
- **Context** Indicates what mode Appraiser is running in. Example: Setup or Diagnostic Data.
- **Time** The client time of the event.
- **AppraiserProcess** The name of the process that launched Appraiser.
- **PCFP** An ID for the system calculated by hashing hardware identifiers.

#### **Microsoft.Windows.Appraiser.General.SetupAdIStatus**

This event indicates if Appraiser used data files from the setup image or more up-to-date data files downloaded from a Microsoft server.

The following fields are available:

- **Time** The client time of the event.

- **PCFP** An ID for the system calculated by hashing hardware identifiers.
- **Result** The last result of the operation to determine if there is a data file to download.
- **OneSettingsInitialized** Was the query to OneSettings, where the information is stored on if there is a data file to download, initialized?
- **Url** The URL of the data file to download. This will be an empty string if there is no data file to download.
- **UsingAlternateData** Is the client using alternate data file or using the data file in the setup image?

#### **Microsoft.Windows.Appraiser.General.SystemMemoryAdd**

This event sends data on the amount of memory on the system and whether it meets requirements, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **Blocking** Is the device from upgrade due to memory restrictions?
- **MemoryRequirementViolated** Was a memory requirement violated?
- **pageFile** The current committed memory limit for the system or the current process, whichever is smaller (in bytes).
- **ram** The amount of memory on the device.
- **ramKB** The amount of memory (in KB).
- **virtual** The size of the user-mode portion of the virtual address space of the calling process (in bytes).
- **virtualKB** The amount of virtual memory (in KB).

#### **Microsoft.Windows.Appraiser.General.SystemMemoryRemove**

This event that the SystemMemory object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemMemoryStartSync**

This event indicates that a new set of SystemMemoryAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorCompareExchangeAdd**

This event sends data indicating whether the system supports the CompareExchange128 CPU requirement, to help keep Windows up to date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **Blocking** Is the upgrade blocked due to the processor?
- **CompareExchange128Support** Does the CPU support CompareExchange128?

#### **Microsoft.Windows.Appraiser.General.SystemProcessorCompareExchangeRemove**

This event indicates that the SystemProcessorCompareExchange object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorCompareExchangeStartSync**

This event indicates that a new set of SystemProcessorCompareExchangeAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorLahfSahfAdd**

This event sends data indicating whether the system supports the LahfSahf CPU requirement, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **Blocking** Is the upgrade blocked due to the processor?
- **LahfSahfSupport** Does the CPU support LAHF/SAHF?

#### **Microsoft.Windows.Appraiser.General.SystemProcessorLahfSahfRemove**

This event indicates that the SystemProcessorLahfSahf object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorLahfSahfStartSync**

This event indicates that a new set of SystemProcessorLahfSahfAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorNxAdd**

This event sends data indicating whether the system supports the NX CPU requirement, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **Blocking** Is the upgrade blocked due to the processor?
- **NXDriverResult** The result of the driver used to do a non-deterministic check for NX support.
- **NXProcessorSupport** Does the processor support NX?

#### **Microsoft.Windows.Appraiser.General.SystemProcessorNxRemove**

This event indicates that the SystemProcessorNx object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorNxStartSync**

This event indicates that a new set of SystemProcessorNxAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorPrefetchWAdd**

This event sends data indicating whether the system supports the PrefetchW CPU requirement, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

- **Blocking** Is the upgrade blocked due to the processor?
- **PrefetchWSupport** Does the processor support PrefetchW?

#### **Microsoft.Windows.Appraiser.General.SystemProcessorPrefetchWRemove**

This event indicates that the SystemProcessorPrefetchW object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorPrefetchWStartSync**

This event indicates that a new set of SystemProcessorPrefetchWAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorSse2Add**

This event sends data indicating whether the system supports the SSE2 CPU requirement, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **Blocking** Is the upgrade blocked due to the processor?
- **SSE2ProcessorSupport** Does the processor support SSE2?

#### **Microsoft.Windows.Appraiser.General.SystemProcessorSse2Remove**

This event indicates that the SystemProcessorSse2 object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemProcessorSse2StartSync**

This event indicates that a new set of SystemProcessorSse2Add events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemTouchAdd**

This event sends data indicating whether the system supports touch, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **IntegratedTouchDigitizerPresent** Is there an integrated touch digitizer?
- **MaximumTouches** The maximum number of touch points supported by the device hardware.

#### **Microsoft.Windows.Appraiser.General.SystemTouchRemove**

This event indicates that the SystemTouch object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemTouchStartSync**

This event indicates that a new set of SystemTouchAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemWimAdd**

This event sends data indicating whether the operating system is running from a compressed WIM file, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **IsWimBoot** Is the current operating system running from a compressed WIM file?
- **RegistryWimBootValue** The raw value from the registry that is used to indicate if the device is running from a WIM.

#### **Microsoft.Windows.Appraiser.General.SystemWimRemove**

This event indicates that the SystemWim object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemWimStartSync**

This event indicates that a new set of SystemWimAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemWindowsActivationStatusAdd**

This event sends data indicating whether the current operating system is activated, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **WindowsIsLicensedApiValue** The result from the API that's used to indicate if operating system is activated.
- **WindowsNotActivatedDecision** Is the current operating system activated?

#### **Microsoft.Windows.Appraiser.General.SystemWindowsActivationStatusRemove**

This event indicates that the SystemWindowsActivationStatus object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemWindowsActivationStatusStartSync**

This event indicates that a new set of SystemWindowsActivationStatusAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemWlanAdd**

This event sends data indicating whether the system has WLAN, and if so, whether it uses an emulated driver that could block an upgrade, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

- **Blocking** Is the upgrade blocked because of an emulated WLAN driver?
- **HasWlanBlock** Does the emulated WLAN driver have an upgrade block?
- **WlanEmulatedDriver** Does the device have an emulated WLAN driver?
- **WlanExists** Does the device support WLAN at all?
- **WlanModulePresent** Are any WLAN modules present?
- **WlanNativeDriver** Does the device have a non-emulated WLAN driver?

#### **Microsoft.Windows.Appraiser.General.SystemWlanRemove**

This event indicates that the SystemWlan object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.SystemWlanStartSync**

This event indicates that a new set of SystemWlanAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.TelemetryRunHealth**

A summary event indicating the parameters and result of a diagnostic data run. This allows the rest of the data sent over the course of the run to be properly contextualized and understood, which is then used to keep Windows up-to-date.

The following fields are available:

- **PerfBackoff** Indicates if the run was invoked with logic to stop running when a user is present. Helps to understand why a run may have a longer elapsed time than normal.
- **RunAppraiser** Indicates if Appraiser was set to run at all. If this is false, it is understood that data events will not be received from this device.
- **ThrottlingUtc** Indicates if the Appraiser client is throttling its output of CUET events to avoid being disabled. This increases runtime but also diagnostic data reliability.
- **AuxInitial** Obsolete, indicates if Appraiser is writing data files to be read by the Get Windows 10 app.
- **Time** The client time of the event.
- **RunDate** The date that the diagnostic data run was stated, expressed as a filetime.
- **AppraiserProcess** The name of the process that launched Appraiser.
- **AppraiserVersion** The file version (major, minor and build) of the Appraiser DLL, concatenated without dots.
- **SendingUtc** Indicates if the Appraiser client is sending events during the current diagnostic data run.
- **DeadlineDate** A timestamp representing the deadline date, which is the time until which appraiser will wait to do a full scan.
- **AppraiserBranch** The source branch in which the version of Appraiser that is running was built.
- **EnterpriseRun** Indicates if the diagnostic data run is an enterprise run, which means appraiser was run from the command line with an extra enterprise parameter.
- **RunGeneralTel** Indicates if the generaltel.dll component was run. Generaltel collects additional diagnostic data on an infrequent schedule and only from machines at diagnostic data levels higher than Basic.
- **PerfBackoffInsurance** Indicates if appraiser is running without performance backoff because it has run with perf backoff and failed to complete several times in a row.
- **AuxFinal** Obsolete, always set to false
- **StoreHandlesNotNull** Obsolete, always set to false
- **VerboseMode** Indicates if appraiser ran in Verbose mode, which is a test-only mode with extra logging.
- **AppraiserDataVersion** The version of the data files being used by the Appraiser diagnostic data run.



- **FullSync** Indicates if Appraiser is performing a full sync, which means that full set of events representing the state of the machine are sent. Otherwise, only the changes from the previous run are sent.
- **InventoryFullSync** Indicates if inventory is performing a full sync, which means that the full set of events representing the inventory of machine are sent.
- **PCFP** An ID for the system calculated by hashing hardware identifiers.
- **RunOnline** Indicates if appraiser was able to connect to Windows Update and therefore is making decisions using up-to-date driver coverage information.
- **TelemetrySent** Indicates if diagnostic data was successfully sent.
- **WhyFullSyncWithoutTablePrefix** Indicates the reason or reasons that a full sync was generated.
- **RunResult** The hresult of the Appraiser diagnostic data run.

#### **Microsoft.Windows.Appraiser.General.WmdrmAdd**

This event sends data about the usage of older digital rights management on the system, to help keep Windows up to date. This data does not indicate the details of the media using the digital rights management, only whether any such files exist. Collecting this data was critical to ensuring the correct mitigation for customers, and should be able to be removed once all mitigations are in place.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **WmdrmCdRipped** Indicates if the system has any files encrypted with personal DRM, which was used for ripped CDs.
- **WmdrmNonPermanent** Indicates if the system has any files with non-permanent licenses.
- **WmdrmPurchased** Indicates if the system has any files with permanent licenses.
- **WmdrmApiResult** Raw value of the API used to gather DRM state.
- **WmdrmInUse** WmdrmIndicators AND dismissible block in setup was not dismissed.
- **WmdrmIndicators** WmdrmCdRipped OR WmdrmPurchased
- **NeedsDismissAction** Indicates if a dismissible message is needed to warn the user about a potential loss of data due to DRM deprecation.
- **BlockingApplication** Same as NeedsDismissAction

#### **Microsoft.Windows.Appraiser.General.WmdrmRemove**

This event indicates that the Wmdrm object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

#### **Microsoft.Windows.Appraiser.General.WmdrmStartSync**

This event indicates that a new set of WmdrmAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

## Census events

### **Census.App**

This event sends version data about the Apps running on this device, to help keep Windows up to date.

The following fields are available:

- **IEVersion** Retrieves which version of Internet Explorer is running on this device.
- **CensusVersion** The version of Census that generated the current data for this device.

## Census.Battery

This event sends type and capacity data about the battery on the device, as well as the number of connected standby devices in use, type to help keep Windows up to date.

The following fields are available:

- **InternalBatteryCapabilities** Represents information about what the battery is capable of doing.
- **InternalBatteryCapacityCurrent** Represents the battery's current fully charged capacity in mWh (or relative). Compare this value to `DesignedCapacity` to estimate the battery's wear.
- **InternalBatteryCapacityDesign** Represents the theoretical capacity of the battery when new, in mWh.
- **IsAlwaysOnAlwaysConnectedCapable** Represents whether the battery enables the device to be `AlwaysOnAlwaysConnected`. Boolean value.
- **InternalBatteryNumberOfCharges** Provides the number of battery charges. This is used when creating new products and validating that existing products meets targeted functionality performance.

## Census.Camera

This event sends data about the resolution of cameras on the device, to help keep Windows up to date.

The following fields are available:

- **FrontFacingCameraResolution** Represents the resolution of the front facing camera in megapixels. If a front facing camera does not exist, then the value is 0.
- **RearFacingCameraResolution** Represents the resolution of the rear facing camera in megapixels. If a rear facing camera does not exist, then the value is 0.

## Census.Enterprise

This event sends data about Azure presence, type, and cloud domain use in order to provide an understanding of the use and integration of devices in an enterprise, cloud, and server environment.

The following fields are available:

- **IsCloudDomainJoined** Is this device joined to an Azure Active Directory (AAD) tenant? true/false
- **IsMDMEnrolled** Whether the device has been MDM Enrolled or not.
- **ServerFeatures** Represents the features installed on a Windows Server. This can be used by developers and administrators who need to automate the process of determining the features installed on a set of server computers.
- **CommercialId** Represents the GUID for the commercial entity which the device is a member of. Will be used to reflect insights back to customers.
- **AzureVMType** Represents whether the instance is Azure VM PAAS, Azure VM IAAS or any other VMs.
- **AzureOSIDPresent** Represents the field used to identify an Azure machine.
- **IsDomainJoined** Indicates whether a machine is joined to a domain.
- **HashedDomain** The hashed representation of the user domain used for login.
- **SystemCenterID** The SCCM ID is an anonymized one-way hash of the Active Directory Organization identifier
- **MPNId** Returns the Partner ID/MPN ID from Regkey.  
HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\DeployID
- **SCCMClientId** This ID correlate systems that send data to Compat Analytics (OMS) and other OMS based systems with systems in an Enterprise SCCM environment.
- **CDJType** Represents the type of cloud domain joined for the machine.
- **IsDeviceProtected** Represents if Device protected by BitLocker/Device Encryption
- **IsDERequirementMet** Represents if the device can do device encryption.
- **IsEDPEnabled** Represents if Enterprise data protected on the device.
- **ContainerType** The type of container, such as process or virtual machine hosted.

- **EnrollmentType** Represents the type of enrollment, such as MDM or Intune, for a particular device.

### Census.Firmware

This event sends data about the BIOS and startup embedded in the device, to help keep Windows up to date.

The following fields are available:

- **FirmwareManufacturer** Represents the manufacturer of the device's firmware (BIOS).
- **FirmwareReleaseDate** Represents the date the current firmware was released.
- **FirmwareType** Represents the firmware type. The various types can be unknown, BIOS, UEFI.
- **FirmwareVersion** Represents the version of the current firmware.

### Census.Fighting

This event sends Windows Insider data from customers participating in improvement testing and feedback programs, to help keep Windows up-to-date.

The following fields are available:

- **FlightIds** A list of the different Windows Insider builds on this device.
- **MSA\_Accounts** Represents a list of hashed IDs of the Microsoft Accounts that are fighting (pre-release builds) on this device.
- **IsFlightsDisabled** Represents if the device is participating in the Windows Insider program.
- **FightingBranchName** The name of the Windows Insider branch currently used by the device.
- **DeviceSampleRate** The diagnostic data sample rate assigned to the device.
- **EnablePreviewBuilds** Used to enable Windows Insider builds on a device.
- **SSRK** Retrieves the mobile targeting settings.

### Census.Hardware

This event sends data about the device, including hardware type, OEM brand, model line, model, diagnostic data level setting, and TPM support, to help keep Windows up-to-date.

The following fields are available:

- **ChassisType** Represents the type of device chassis, such as desktop or low profile desktop. The possible values can range between 1 - 36.
- **ComputerHardwareID** Identifies a device class that is represented by a hash of different SMBIOS fields.
- **DeviceColor** Indicates a color of the device.
- **DeviceName** The device name that is set by the user.
- **OEMDigitalMarkerFileName** The name of the file placed in the \Windows\system32\drivers directory that specifies the OEM and model name of the device.
- **OEMManufacturerName** The device manufacturer name. The OEMName for an inactive device is not reprocessed even if the clean OEM name is changed at a later date.
- **OEMModelNumber** The device model number.
- **OEMModelName** The device model name.
- **OEMModelSKU** The device edition that is defined by the manufacturer.
- **OEMOptionalIdentifier** A Microsoft assigned value that represents a specific OEM subsidiary.
- **OEMSerialNumber** The serial number of the device that is set by the manufacturer.
- **PhoneManufacturer** The friendly name of the phone manufacturer.
- **SoCName** The firmware manufacturer of the device.
- **DUID** The device unique ID.
- **InventoryId** The device ID used for compatibility testing.
- **VoiceSupported** Does the device have a cellular radio capable of making voice calls?

- **PowerPlatformRole** The OEM preferred power management profile. It's used to help to identify the basic form factor of the device.
- **TPMVersion** The supported Trusted Platform Module (TPM) on the device. If no TPM is present, the value is 0.
- **StudyID** Used to identify retail and non-retail device.
- **TelemetryLevel** The diagnostic data level the user has opted into, such as Basic or Enhanced.
- **TelemetrySettingAuthority** Determines who set the diagnostic data level, such as GP, MDM, or the user.
- **DeviceForm** Indicates the form as per the device classification.
- **DigitizerSupport** Is a digitizer supported?
- **OEMModelBaseBoard** The baseboard model used by the OEM.
- **OEMModelSystemFamily** The system family set on the device by an OEM.
- **OEMModelBaseBoardVersion** Differentiates between developer and retail devices.
- **ActiveMicCount** The number of active microphones attached to the device.
- **OEMModelSystemVersion** The system model version set on the device by the OEM.
- **D3DMaxFeatureLevel** The supported Direct3D version.
- **Gyroscope** Indicates whether the device has a gyroscope.
- **Magnetometer** Indicates whether the device has a magnetometer.
- **NFCProximity** Indicates whether the device supports NFC.
- **TelemetryLevelLimitEnhanced** The diagnostic data level for Windows Analytics-based solutions.

### Census.Memory

This event sends data about the memory on the device, including ROM and RAM, to help keep Windows up to date.

The following fields are available:

- **TotalPhysicalRAM** Represents the physical memory (in MB).
- **TotalVisibleMemory** Represents the memory that is not reserved by the system.

### Census.Network

This event sends data about the mobile and cellular network used by the device (mobile service provider, network, device ID, and service cost factors), to help keep Windows up to date.

The following fields are available:

- **MobileOperatorBilling** Represents the telephone company that provides services for mobile phone users.
- **MobileOperatorCommercialized** Represents which reseller and geography the phone is commercialized for. This is the set of values on the phone for who and where it was intended to be used. For example, the commercialized mobile operator code AT&T in the US would be ATT-US.
- **NetworkCost** Represents the network cost associated with a connection.
- **IMEI0** Represents the International Mobile Station Equipment Identity. This number is usually unique and used by the mobile operator to distinguish different phone hardware. Microsoft does not have access to mobile operator billing data so collecting this data does not expose or identify the user. The two fields represent phone with dual sim coverage.
- **SPN0** Retrieves the Service Provider Name (SPN). For example, these might be AT&T, Sprint, T-Mobile, or Verizon. The two fields represent phone with dual sim coverage.
- **MobileOperatorNetwork0** Represents the operator of the current mobile network that the device is used on. (AT&T, T-Mobile, Vodafone). The two fields represent phone with dual sim coverage.
- **MCC0** Represents the Mobile Country Code (MCC). It used with the Mobile Network Code (MNC) to uniquely identify a mobile network operator. The two fields represent phone with dual sim coverage.
- **MNC0** Retrieves the Mobile Network Code (MNC). It used with the Mobile Country Code (MCC) to uniquely identify a mobile network operator. The two fields represent phone with dual sim coverage.

- **IMEI1** Represents the International Mobile Station Equipment Identity. This number is usually unique and used by the mobile operator to distinguish different phone hardware. Microsoft does not have access to mobile operator billing data so collecting this data does not expose or identify the user. The two fields represent phone with dual sim coverage.
- **SPN1** Retrieves the Service Provider Name (SPN). For example, these might be AT&T, Sprint, T-Mobile, or Verizon. The two fields represent phone with dual sim coverage.
- **MobileOperatorNetwork1** Represents the operator of the current mobile network that the device is used on. (AT&T, T-Mobile, Vodafone). The two fields represent phone with dual sim coverage.
- **MCC1** Represents the Mobile Country Code (MCC). It used with the Mobile Network Code (MNC) to uniquely identify a mobile network operator. The two fields represent phone with dual sim coverage.
- **MNC1** Retrieves the Mobile Network Code (MNC). It used with the Mobile Country Code (MCC) to uniquely identify a mobile network operator. The two fields represent phone with dual sim coverage.
- **MEID** Represents the Mobile Equipment Identity (MEID). MEID is a worldwide unique phone ID assigned to CDMA phones. MEID replaces electronic serial number (ESN), and is equivalent to IMEI for GSM and WCDMA phones. Microsoft does not have access to mobile operator billing data so collecting this data does not expose or identify the user.
- **NetworkAdapterGUID** The GUID of the primary network adapter.

### Census.OS

This event sends data about the operating system such as the version, locale, update service configuration, when and how it was originally installed, and whether it is a virtual device, to help keep Windows up to date.

The following fields are available:

- **GenuineState** Retrieves the ID Value specifying the OS Genuine check.
- **IsPortableOperatingSystem** Retrieves whether OS is running Windows-To-Go
- **IsSecureBootEnabled** Retrieves whether Boot chain is signed under UEFI.
- **InstallationType** Retrieves the type of OS installation. (Clean, Upgrade, Reset, Refresh, Update).
- **OSInstallType** Retrieves a numeric description of what install was used on the device i.e. clean, upgrade, refresh, reset, etc
- **OSOBEDateTime** Retrieves Out of Box Experience (OOBE) Date in Coordinated Universal Time (UTC).
- **OSSKU** Retrieves the Friendly Name of OS Edition.
- **OSTimeZoneBiasInMins** Retrieves the time zone set on machine.
- **OSUILocale** Retrieves the locale of the UI that is currently used by the OS.
- **RACw7Id** Retrieves the Microsoft Reliability Analysis Component (RAC) Win7 Identifier. RAC is used to monitor and analyze system usage and reliability.
- **CompactOS** Indicates if the Compact OS feature from Win10 is enabled.
- **Signature** Retrieves if it is a signature machine sold by Microsoft store.
- **IsDeviceRetailDemo** Retrieves if the device is running in demo mode.
- **ActivationChannel** Retrieves the retail license key or Volume license key for a machine.
- **LicenseStateReason** Retrieves why (or how) a system is licensed or unlicensed. The HRESULT may indicate an error code that indicates a key blocked error, or it may indicate that we are running an OS License granted by the MS store.
- **OA3xOriginalProductKey** Retrieves the License key stamped by the OEM to the machine.
- **ProductKeyID2** Retrieves the License key if the machine is updated with a new license key.
- **ServiceMachineIP** Retrieves the IP address of the KMS host used for anti-piracy.
- **ServiceProductKeyID** Retrieves the License key of the KMS
- **LanguagePacks** The list of language packages installed on the device.
- **InstallLanguage** The first language installed on the user machine.
- **IsEduData** Returns Boolean if the education data policy is enabled.

- **SharedPCMode** Returns Boolean for education devices used as shared cart
- **SLICVersion** Returns OS type/version from SLIC table.
- **SLICStatus** Whether a SLIC table exists on the device.
- **OSEdition** Retrieves the version of the current OS.
- **ProductActivationTime** Returns the OS Activation time for tracking piracy issues.
- **ProductActivationResult** Returns Boolean if the OS Activation was successful.
- **OSSubscriptionTypeIeld** Returns boolean for enterprise subscription feature for selected PRO machines.
- **OSSubscriptionStatus** Represents the existing status for enterprise subscription feature for PRO machines.
- **ServiceMachinePort** Retrieves the port of the KMS host used for anti-piracy.
- **DeviceTimeZone** The time zone that is set on the device. Example: Pacific Standard Time
- **DeveloperUnlockStatus** Represents if a device has been developer unlocked by the user or Group Policy.
- **AssignedAccessStatus** The kiosk configuration mode.

### Census.Processor

This event sends data about the processor (architecture, speed, number of cores, manufacturer, and model number), to help keep Windows up to date.

The following fields are available:

- **KvaShadow** Microcode info of the processor.
- **MMSettingOverride** Microcode setting of the processor.
- **MMSettingOverrideMask** Microcode setting override of the processor.
- **ProcessorArchitecture** Retrieves the processor architecture of the installed operating system.
- **ProcessorClockSpeed** Retrieves the clock speed of the processor in MHz.
- **ProcessorCores** Retrieves the number of cores in the processor.
- **ProcessorIdentifier** The processor identifier of a manufacturer.
- **ProcessorManufacturer** Retrieves the name of the processor's manufacturer.
- **ProcessorModel** Retrieves the name of the processor model.
- **ProcessorPhysicalCores** Number of physical cores in the processor.
- **ProcessorUpdateRevision** The microcode version.
- **SocketCount** Number of physical CPU sockets of the machine.
- **SpeculationControl** If the system has enabled protections needed to validate the speculation control vulnerability.

### Census.Speech

This event is used to gather basic speech settings on the device.

The following fields are available:

- **AboveLockEnabled** Cortana setting that represents if Cortana can be invoked when the device is locked.
- **GPAllowInputPersonalization** Indicates if a Group Policy setting has enabled speech functionalities.
- **HolographicSpeechInputDisabled** Holographic setting that represents if the attached HMD devices have speech functionality disabled by the user.
- **HolographicSpeechInputDisabledRemote** Indicates if a remote policy has disabled speech functionalities for the HMD devices.
- **KWSEnabled** Cortana setting that represents if a user has enabled the "Hey Cortana" keyword spotter (KWS).
- **MDMAllowInputPersonalization** Indicates if an MDM policy has enabled speech functionalities.
- **RemotelyManaged** Indicates if the device is being controlled by a remote administrator (MDM or Group Policy) in the context of speech functionalities.
- **SpeakerIdEnabled** Cortana setting that represents if keyword detection has been trained to try to respond to a single user's voice.

- **SpeechServicesEnabled** Windows setting that represents whether a user is opted-in for speech services on the device.

### Census.Storage

This event sends data about the total capacity of the system volume and primary disk, to help keep Windows up to date.

The following fields are available:

- **PrimaryDiskTotalCapacity** Retrieves the amount of disk space on the primary disk of the device in MB.
- **SystemVolumeTotalCapacity** Retrieves the size of the partition that the System volume is installed on in MB.
- **PrimaryDiskType** Retrieves an enumerator value of type STORAGE\_BUS\_TYPE that indicates the type of bus to which the device is connected. This should be used to interpret the raw device properties at the end of this structure (if any).

### Census.Userdefault

This event sends data about the current user's default preferences for browser and several of the most popular extensions and protocols, to help keep Windows up to date.

The following fields are available:

- **DefaultBrowserProgId** The ProgramId of the current user's default browser
- **DefaultApp** The current user's default program selected for the following extension or protocol:  
.html,.htm,.jpg,.jpeg,.png,.mp3,.mp4, .mov,.pdf

### Census.UserDisplay

This event sends data about the logical/physical display size, resolution and number of internal/external displays, and VRAM on the system, to help keep Windows up to date.

The following fields are available:

- **InternalPrimaryDisplayLogicalDPIX** Retrieves the logical DPI in the x-direction of the internal display.
- **InternalPrimaryDisplayLogicalDPIY** Retrieves the logical DPI in the y-direction of the internal display.
- **InternalPrimaryDisplayPhysicalDPIX** Retrieves the physical DPI in the x-direction of the internal display.
- **InternalPrimaryDisplayPhysicalDPIY** Retrieves the physical DPI in the y-direction of the internal display.
- **InternalPrimaryDisplayResolutionHorizontal** Retrieves the number of pixels in the horizontal direction of the internal display.
- **InternalPrimaryDisplayResolutionVertical** Retrieves the number of pixels in the vertical direction of the internal display.
- **InternalPrimaryDisplaySizePhysicalH** Retrieves the physical horizontal length of the display in mm. Used for calculating the diagonal length in inches .
- **InternalPrimaryDisplaySizePhysicalY** Retrieves the physical vertical length of the display in mm. Used for calculating the diagonal length in inches
- **NumberOfInternalDisplays** Retrieves the number of internal displays in a machine.
- **NumberOfExternalDisplays** Retrieves the number of external displays connected to the machine
- **VRAMDedicated** Retrieves the video RAM in MB.
- **VRAMDedicatedSystem** Retrieves the amount of memory on the dedicated video card.
- **VRAMSharedSystem** Retrieves the amount of RAM memory that the video card can use.

### Census.UserNLS

This event sends data about the default app language, input, and display language preferences set by the user, to help keep Windows up to date.

The following fields are available:

- **DefaultAppLanguage** The current user Default App Language.
- **HomeLocation** The current user location, which is populated using GetUserGeold() function.
- **DisplayLanguage** The current user preferred Windows Display Language.
- **SpeechInputLanguages** The Speech Input languages installed on the device.
- **KeyboardInputLanguages** The Keyboard input languages installed on the device.

### Census.VM

This event sends data indicating whether virtualization is enabled on the device, and its various characteristics, to help keep Windows up to date.

The following fields are available:

- **VirtualizationFirmwareEnabled** Represents whether virtualization is enabled in the firmware.
- **SLATSupported** Represents whether Second Level Address Translation (SLAT) is supported by the hardware.
- **IOMMUPresent** Represents if an input/output memory management unit (IOMMU) is present.
- **IsVirtualDevice** Retrieves that when the Hypervisor is Microsoft's Hyper-V Hypervisor or other Hv#1 Hypervisor, this field will be set to FALSE for the Hyper-V host OS and TRUE for any guest OS's. This field should not be relied upon for non-Hv#1 Hypervisors.
- **HyperVisor** Retrieves whether the current OS is running on top of a Hypervisor.
- **CloudService** Indicates which cloud service, if any, that this virtual machine is running within.
- **isVDI** Is the device using Virtual Desktop Infrastructure?

### Census.WU

This event sends data about the Windows update server and other App store policies, to help keep Windows up to date.

The following fields are available:

- **WUMachineId** Retrieves the Windows Update (WU) Machine Identifier.
- **WU Server** Retrieves the HTTP(S) URL of the WSUS server that is used by Automatic Updates and API callers (by default).
- **WUDODownloadMode** Retrieves whether DO is turned on and how to acquire/distribute updates Delivery Optimization (DO) allows users to deploy previously downloaded WU updates to other devices on the same network.
- **OSWUAutoUpdateOptions** Retrieves the auto update settings on the device.
- **AppStoreAutoUpdate** Retrieves the Appstore settings for auto upgrade. (Enable/Disabled).
- **AppStoreAutoUpdatePolicy** Retrieves the Microsoft Store App Auto Update group policy setting
- **AppStoreAutoUpdateMDM** Retrieves the App Auto Update value for MDM: 0 - Disallowed. 1 - Allowed. 2 - Not configured. Default: [2] Not configured
- **DelayUpgrade** Retrieves the Windows upgrade flag for delaying upgrades.
- **UpdateServiceURLConfigured** Retrieves if the device is managed by Windows Server Update Services (WSUS).
- **WUDeferUpgradePeriod** Retrieves if deferral is set for Upgrades
- **WUDeferUpdatePeriod** Retrieves if deferral is set for Updates
- **WUPauseState** Retrieves WU setting to determine if updates are paused
- **OSUninstalled** A flag that represents when a feature update is uninstalled on a device .
- **OSRolledBack** A flag that represents when a feature update has rolled back during setup.
- **OSRollbackCount** The number of times feature updates have rolled back on the device.
- **UninstallActive** A flag that represents when a device has uninstalled a previous upgrade recently.
- **AppraiserGatedStatus** Indicates whether a device has been gated for upgrading.
- **OSAssessmentFeatureOutOfDate** How many days has it been since a the last feature update was released



but the device did not install it?

- **OSAssessmentForFeatureUpdate** Is the device is on the latest feature update?
- **OSAssessmentForQualityUpdate** Is the device on the latest quality update?
- **OSAssessmentForSecurityUpdate** Is the device on the latest security update?
- **OSAssessmentQualityOutOfDate** How many days has it been since a the last quality update was released but the device did not install it?
- **OSAssessmentReleaseInfoTime** The freshness of release information used to perform an assessment.

### Census.Xbox

This event sends data about the Xbox Console, such as Serial Number and DeviceId, to help keep Windows up to date.

The following fields are available:

- **XboxLiveDeviceId** Retrieves the unique device id of the console.
- **XboxConsoleSerialNumber** Retrieves the serial number of the Xbox console.
- **XboxLiveSandboxId** Retrieves the developer sandbox id if the device is internal to MS.
- **XboxConsolePreferredLanguage** Retrieves the preferred language selected by the user on Xbox console.

### Census.Security

This event provides information on about security settings used to help keep Windows up-to-date and secure.

- **AvailableSecurityProperties** Enumerates and reports state on the relevant security properties for Device Guard.
- **CGRunning** Is Credential Guard running?
- **DGState** A summary of the Device Guard state.
- **HVCIRunning** Is HVCI running?
- **RequiredSecurityProperties** Describes the required security properties to enable virtualization-based security.
- **SecureBootCapable** Is this device capable of running Secure Boot?
- **VBSState** Is virtualization-based security enabled, disabled, or running?

## Diagnostic data events

### TelClientSynthetic.AuthorizationInfo\_RuntimeTransition

This event sends data indicating that a device has undergone a change of diagnostic data opt-in level during the runtime of the device (not at UTC boot or offline), to help keep Windows up to date.

The following fields are available:

- **CanAddMsaToMsTelemetry** True if UTC is allowed to add MSA user identity onto diagnostic data from the OS provider groups.
- **CanCollectAnyTelemetry** True if UTC is allowed to collect non-OS diagnostic data. Non-OS diagnostic data is responsible for providing its own opt-in mechanism.
- **CanCollectCoreTelemetry** True if UTC is allowed to collect data which is tagged with both MICROSOFT\_KEYWORD\_CRITICAL\_DATA and MICROSOFT\_EVENTTAG\_CORE\_DATA.
- **CanCollectHeartbeats** True if UTC is allowed to collect heartbeats.
- **CanCollectOsTelemetry** True if UTC is allowed to collect diagnostic data from the OS provider groups.
- **CanPerformDiagnosticEscalations** True if UTC is allowed to perform all scenario escalations.
- **CanPerformScripting** True if UTC is allowed to perform scripting.
- **CanPerformTraceEscalations** True if UTC is allowed to perform scenario escalations with tracing actions.
- **CanReportScenarios** True if UTC is allowed to load and report scenario completion, failure, and cancellation

events.

- **TransitionFromEverythingOff** True if this transition is moving from not allowing core diagnostic data to allowing core diagnostic data.
- **PreviousPermissions** Bitmask representing the previously configured permissions since the diagnostic data opt-in level was last changed.

### **TelClientSynthetic.AuthorizationInfo\_Startup**

This event sends data indicating that a device has undergone a change of diagnostic data opt-in level detected at UTC startup, to help keep Windows up to date.

The following fields are available:

- **TransitionFromEverythingOff** True if this transition is moving from not allowing core diagnostic data to allowing core diagnostic data.
- **CanCollectAnyTelemetry** True if UTC is allowed to collect non-OS diagnostic data. Non-OS diagnostic data is responsible for providing its own opt-in mechanism.
- **CanCollectHeartbeats** True if UTC is allowed to collect heartbeats.
- **CanCollectCoreTelemetry** True if UTC is allowed to collect data which is tagged with both MICROSOFT\_KEYWORD\_CRITICAL\_DATA and MICROSOFT\_EVENTTAG\_CORE\_DATA.
- **CanCollectOsTelemetry** True if UTC is allowed to collect diagnostic data from the OS provider groups.
- **CanReportScenarios** True if UTC is allowed to load and report scenario completion, failure, and cancellation events.
- **CanAddMsaToMsTelemetry** True if UTC is allowed to add MSA user identity onto diagnostic data from the OS provider groups.
- **CanPerformTraceEscalations** True if UTC is allowed to perform scenario escalations with tracing actions.
- **CanPerformDiagnosticEscalations** True if UTC is allowed to perform all scenario escalations.
- **CanPerformScripting** True if UTC is allowed to perform scripting.
- **PreviousPermissions** Bitmask representing the previously configured permissions since the diagnostic data client was last started.

### **TelClientSynthetic.ConnectivityHeartBeat\_0**

This event sends data about the connectivity status of the Connected User Experiences and Telemetry component that uploads diagnostic data events. If an unrestricted free network (such as Wi-Fi) is available, this event updates the last successful upload time. Otherwise, it checks whether a Connectivity Heartbeat event was fired in the past 24 hours, and if not, it fires an event. A Connectivity Heartbeat event also fires when a device recovers from costed network to free network.

The following fields are available:

- **CensusExitCode** Returns last execution codes from census client run.
- **CensusStartTime** Returns timestamp corresponding to last successful census run.
- **CensusTaskEnabled** Returns Boolean value for the census task (Enable/Disable) on client machine.
- **LastConnectivityLossTime** Retrieves the last time the device lost free network.
- **NetworkState** Retrieves the network state: 0 = No network. 1 = Restricted network. 2 = Free network.
- **NoNetworkTime** Retrieves the time spent with no network (since the last time) in seconds.
- **RestrictedNetworkTime** Retrieves the time spent on a metered (cost restricted) network in seconds.
- **LastConnectivityLossTime** Retrieves the last time the device lost free network.

### **TelClientSynthetic.HeartBeat\_5**

This event sends data about the health and quality of the diagnostic data from the given device, to help keep Windows up to date. It also enables data analysts to determine how 'trusted' the data is from a given device.

The following fields are available:

- **PreviousHeartBeatTime** The time of last heartbeat event. This allows chaining of events.
- **EtwDroppedCount** The number of events dropped by the ETW layer of the diagnostic data client.
- **ConsumerDroppedCount** The number of events dropped by the consumer layer of the diagnostic data client.
- **DecodingDroppedCount** The number of events dropped because of decoding failures.
- **ThrottledDroppedCount** The number of events dropped due to throttling of noisy providers.
- **DbDroppedCount** The number of events that were dropped because the database was full.
- **EventSubStoreResetCounter** The number of times the event database was reset.
- **EventSubStoreResetSizeSum** The total size of the event database across all resets reports in this instance.
- **CriticalOverflowEntersCounter** The number of times a critical overflow mode was entered into the event database.
- **EnteringCriticalOverflowDroppedCounter** The number of events that was dropped because a critical overflow mode was initiated.
- **UploaderDroppedCount** The number of events dropped by the uploader layer of the diagnostic data client.
- **InvalidHttpCodeCount** The number of invalid HTTP codes received from Vortex.
- **LastInvalidHttpCode** The last invalid HTTP code received from Vortex.
- **MaxInUseScenarioCounter** The soft maximum number of scenarios loaded by the Connected User Experiences and Telemetry component.
- **LastEventSizeOffender** The name of the last event that exceeded the maximum event size.
- **SettingsHttpAttempts** The number of attempts to contact the OneSettings service.
- **SettingsHttpFailures** The number of failures from contacting the OneSettings service.
- **VortexHttpAttempts** The number of attempts to contact the Vortex service.
- **EventsUploaded** The number of events that have been uploaded.
- **DbCriticalDroppedCount** The total number of dropped critical events in the event database.
- **VortexHttpFailures4xx** The number of 400-499 error codes received from Vortex.
- **VortexHttpFailures5xx** The number of 500-599 error codes received from Vortex.
- **VortexFailuresTimeout** The number of timeout failures received from Vortex.
- **HeartBeatSequenceNumber** A monotonically increasing heartbeat counter.
- **EtwDroppedBufferCount** The number of buffers dropped in the CUET ETW session.
- **FullTriggerBufferDroppedCount** The number of events that were dropped because the trigger buffer was full.
- **CriticalDataThrottleDroppedCount** The number of critical data sampled events that were dropped because of throttling.
- **CriticalDataDbDroppedCount** The number of critical data sampled events that were dropped at the database layer.
- **MaxActiveAgentConnectionCount** The maximum number of active agents during this heartbeat timeframe.
- **AgentConnectionErrorsCount** The number of non-timeout errors associated with the host/agent channel.
- **LastAgentConnectionError** The last non-timeout error that happened in the host/agent channel.
- **Flags** Flags that indicate device state, such as network, battery, and opt-in state.
- **CensusTaskEnabled** Indicates whether Census is enabled.
- **CensusExitCode** The last exit code of the Census task.
- **CensusStartTime** The time of the last Census run.

#### TelClientSynthetic.PrivacySettingsAfterCreatorsUpdate

This event sends basic data on privacy settings before and after a feature update. This is used to ensure that customer privacy settings are correctly migrated across feature updates.

The following fields are available:

- **PostUpgradeSettings** The privacy settings after a feature update.

- **PreUpgradeSettings** The privacy settings before a feature update.

## DxgKernelTelemetry events

### DxgKrnlTelemetry.GPUAdapterInventoryV2

This event sends basic GPU and display driver information to keep Windows and display drivers up-to-date.

The following fields are available:

- **version** The event version.
- **bootId** The system boot ID.
- **aiSeqId** The event sequence ID.
- **MeasureEnabled** Is the device listening to MICROSOFT\_KEYWORD\_MEASURES?
- **TelemetryEnabled** Is the device listening to MICROSOFT\_KEYWORD\_TELEMETRY?
- **InterfaceId** The GPU interface ID.
- **GPUVendorID** The GPU vendor ID.
- **GPUDeviceID** The GPU device ID.
- **SubVendorID** The GPU sub vendor ID.
- **SubSystemID** The subsystem ID.
- **GPURevisionID** The GPU revision ID.
- **DriverVersion** The display driver version.
- **DriverDate** The date of the display driver.
- **DriverRank** The rank of the display driver.
- **IsMiracastSupported** Does the GPU support Miracast?
- **IsMsMiracastSupported** Are the GPU Miracast capabilities driven by a Microsoft solution?
- **IsHybridDiscrete** Does the GPU have discrete GPU capabilities in a hybrid device?
- **IsHybridIntegrated** Does the GPU have integrated GPU capabilities in a hybrid device?
- **IsMPOSupported** Does the GPU support Multi-Plane Overlays?
- **IsLDA** Is the GPU comprised of Linked Display Adapters?
- **IsMismatchLDA** Is at least one device in the Linked Display Adapters chain from a different vendor?
- **IsPostAdapter** Is this GPU the POST GPU in the device?
- **IsSoftwareDevice** Is this a software implementation of the GPU?
- **IsRenderDevice** Does the GPU have rendering capabilities?
- **IsDisplayDevice** Does the GPU have displaying capabilities?
- **WDDMVersion** The Windows Display Driver Model version.
- **DisplayAdapterLuid** The display adapter LUID.
- **GPUPreemptionLevel** The maximum preemption level supported by GPU for graphics payload.
- **ComputePreemptionLevel** The maximum preemption level supported by GPU for compute payload.
- **TellInvEvtTrigger** What triggered this event to be logged? Example: 0 (GPU enumeration) or 1 (DxgKrnlTelemetry provider toggling)
- **DedicatedVideoMemoryB** The amount of dedicated VRAM of the GPU (in bytes).
- **DedicatedSystemMemoryB** The amount of system memory dedicated for GPU use (in bytes).
- **SharedSystemMemoryB** The amount of system memory shared by GPU and CPU (in bytes).
- **NumVidPnSources** The number of supported display output sources.
- **NumVidPnTargets** The number of supported display output targets.

## Fault Reporting events

### Microsoft.Windows.FaultReporting.AppCrashEvent

This event sends data about crashes for both native and managed applications, to help keep Windows up to date. The data includes information about the crashing process and a summary of its exception record. It does not contain any Watson bucketing information. The bucketing information is recorded in a Windows Error Reporting (WER) event that is generated when the WER client reports the crash to the Watson service, and the WER event will contain the same ReportID (see field 14 of crash event, field 19 of WER event) as the crash event for the crash being reported. AppCrash is emitted once for each crash handled by WER (e.g. from an unhandled exception or FailFast or ReportException). Note that Generic Watson event types (e.g. from PLM) that may be considered crashes" by a user DO NOT emit this event.

The following fields are available:

- **ProcessId** The ID of the process that has crashed.
- **ProcessCreateTime** The time of creation of the process that has crashed.
- **ExceptionCode** The exception code returned by the process that has crashed.
- **ExceptionOffset** The address where the exception had occurred.
- **AppName** The name of the app that has crashed.
- **AppVersion** The version of the app that has crashed.
- **AppTimeStamp** The date/time stamp of the app.
- **ModName** Exception module name (e.g. bar.dll).
- **ModVersion** The version of the module that has crashed.
- **ModTimeStamp** The date/time stamp of the module.
- **PackageFullName** Store application identity.
- **PackageRelativeAppId** Store application identity.
- **ProcessArchitecture** Architecture of the crashing process, as one of the PROCESSOR\_ARCHITECTURE\_\* constants: 0: PROCESSOR\_ARCHITECTURE\_INTEL. 5: PROCESSOR\_ARCHITECTURE\_ARM. 9: PROCESSOR\_ARCHITECTURE\_AMD64. 12: PROCESSOR\_ARCHITECTURE\_ARM64.
- **ReportId** A GUID used to identify the report. This can be used to track the report across Watson.
- **Flags** Flags indicating how reporting is done. For example, queue the report, do not offer JIT debugging, or do not terminate the process after reporting.
- **AppSessionGuid** GUID made up of process ID and is used as a correlation vector for process instances in the diagnostic data backend.
- **TargetAppId** The kernel reported AppId of the application being reported.
- **TargetAppVer** The specific version of the application being reported
- **TargetAsId** The sequence number for the hanging process.

## Hang Reporting events

### Microsoft.Windows.HangReporting.AppHangEvent

This event sends data about hangs for both native and managed applications, to help keep Windows up to date. It does not contain any Watson bucketing information. The bucketing information is recorded in a Windows Error Reporting (WER) event that is generated when the WER client reports the hang to the Watson service, and the WER event will contain the same ReportID (see field 13 of hang event, field 19 of WER event) as the hang event for the hang being reported. AppHang is reported only on PC devices. It handles classic Win32 hangs and is emitted only once per report. Some behaviors that may be perceived by a user as a hang are reported by app managers (e.g. PLM/RM/EM) as Watson Generics and will not produce AppHang events.

The following fields are available:

- **AppName** The name of the app that has hung.
- **TypeCode** Bitmap describing the hang type.
- **ProcessId** The ID of the process that has hung.

- **UTCReplace\_TargetAppId** The kernel reported AppId of the application being reported.
- **ProcessCreateTime** The time of creation of the process that has hung.
- **UTCReplace\_TargetAppVer** The specific version of the application being reported.
- **WaitingOnAppName** If this is a cross process hang waiting for an application, this has the name of the application.
- **PackageRelativeAppId** Store application identity.
- **ProcessArchitecture** Architecture of the hung process, as one of the PROCESSOR\_ARCHITECTURE\_\* constants: 0: PROCESSOR\_ARCHITECTURE\_INTEL. 5: PROCESSOR\_ARCHITECTURE\_ARM. 9: PROCESSOR\_ARCHITECTURE\_AMD64. 12: PROCESSOR\_ARCHITECTURE\_ARM64.
- **WaitingOnPackageRelativeAppId** If this is a cross process hang waiting for a package, this has the relative application id of the package.
- **WaitingOnAppVersion** If this is a cross process hang, this has the version of the application for which it is waiting.
- **AppSessionGuid** GUID made up of process id used as a correlation vector for process instances in the diagnostic data backend.
- **WaitingOnPackageFullName** If this is a cross process hang waiting for a package, this has the full name of the package for which it is waiting.
- **PackageFullName** Store application identity.
- **AppVersion** The version of the app that has hung.
- **ReportId** A GUID used to identify the report. This can be used to track the report across Watson.
- **TargetAppId** The kernel reported AppId of the application being reported.
- **TargetAppVer** The specific version of the application being reported.
- **TargetAsId** The sequence number for the hanging process.

## Inventory events

### Microsoft.Windows.Inventory.Core.AmiTelCacheChecksum

This event captures basic checksum data about the device inventory items stored in the cache for use in validating data completeness for Microsoft.Windows.Inventory.Core events. The fields in this event may change over time, but they will always represent a count of a given object.

The following fields are available:

- **Device** A count of device objects in cache
- **DeviceCensus** A count of devicecensus objects in cache
- **DriverPackageExtended** A count of driverpackageextended objects in cache
- **File** A count of file objects in cache
- **Generic** A count of generic objects in cache
- **HwItem** A count of hwitem objects in cache
- **InventoryApplication** A count of application objects in cache
- **InventoryApplicationFile** A count of application file objects in cache
- **InventoryDeviceContainer** A count of device container objects in cache
- **InventoryDeviceMediaClass** A count of device media objects in cache
- **InventoryDevicePnp** A count of devicepnp objects in cache
- **InventoryDriverBinary** A count of driver binary objects in cache
- **InventoryDriverPackage** A count of device objects in cache
- **Metadata** A count of metadata objects in cache
- **Orphan** A count of orphan file objects in cache
- **Programs** A count of program objects in cache

- **FileSigningInfo** A count of file signing info objects in cache.
- **InventoryDeviceInterface** A count of inventory device interface objects in cache.

#### **Microsoft.Windows.Inventory.Core.AmiTelCacheVersions**

This event sends inventory component versions for the Device Inventory data.

The following fields are available:

- **aeinv** The version of the App inventory component.
- **devinv** The file version of the Device inventory component.

#### **Microsoft.Windows.Inventory.Core.InventoryDeviceUsbHubClassStartSync**

This event indicates that a new set of InventoryDeviceUsbHubClassAdd events will be sent

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events
- 

#### **Microsoft.Windows.Inventory.Core.InventoryDeviceUsbHubClassAdd**

This event sends basic metadata about the USB hubs on the device

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events
- **TotalUserConnectablePorts** Total number of connectable USB ports
- **TotalUserConnectableTypeCPorts** Total number of connectable USB Type C ports
- 

#### **Microsoft.Windows.Inventory.Core.InventoryApplicationAdd**

This event sends basic metadata about an application on the system to help keep Windows up to date.

The following fields are available:

- **ProgramInstanceId** A hash of the file IDs in an app.
- **Name** The name of the application. Location pulled from depends on 'Source' field.
- **Type** One of ("Application", "Hotfix", "BOE", "Service", "Unknown"). Application indicates Win32 or Appx app, Hotfix indicates app updates (KBs), BOE indicates it's an app with no ARP or MSI entry, Service indicates that it is a service. Application and BOE are the ones most likely seen.
- **Publisher** The Publisher of the application. Location pulled from depends on the 'Source' field.
- **Version** The version number of the program.
- **Language** The language code of the program.
- **Source** How the program was installed (ARP, MSI, Appx, etc...)
- **MsiProductCode** A GUID that describe the MSI Product.
- **MsiPackageCode** A GUID that describes the MSI Package. Multiple 'Products' (apps) can make up an MsiPackage.
- **HiddenArp** Indicates whether a program hides itself from showing up in ARP.
- **OSVersionAtInstallTime** The four octets from the OS version at the time of the application's install.
- **RootDirPath** The path to the root directory where the program was installed.
- **InstallDate** The date the application was installed (a best guess based on folder creation date heuristics)
- **InstallDateMsi** The install date if the application was installed via MSI. Passed as an array.
- **InstallDateFromLinkFile** The estimated date of install based on the links to the files. Passed as an array.
- **InstallDateArpLastModified** The date of the registry ARP key for a given application. Hints at install date but not always accurate. Passed as an array.

- **PartB\_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **objectInstanceid** ProgramId (a hash of Name, Version, Publisher, and Language of an application used to identify it).
- **PackageFullName** The package full name for a Store application.
- **InventoryVersion** The version of the inventory file generating the events.
- **StoreAppType** A sub-classification for the type of Microsoft Store app, such as UWP or Win8StoreApp.

#### **Microsoft.Windows.Inventory.Core.InventoryApplicationRemove**

This event indicates that a new set of InventoryDevicePnpAdd events will be sent.

The following fields are available:

- **PartB\_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryApplicationStartSync**

This event indicates that a new set of InventoryApplicationAdd events will be sent.

The following fields are available:

- **PartB\_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDeviceContainerAdd**

This event sends basic metadata about a device container (such as a monitor or printer as opposed to a PNP device) to help keep Windows up-to-date.

The following fields are available:

- **ModelName** The model name.
- **ModelId** A model GUID.
- **PrimaryCategory** The primary category for the device container.
- **Categories** A comma separated list of functional categories in which the container belongs.
- **IsConnected** For a physically attached device, this value is the same as IsPresent. For wireless a device, this value represents a communication link.
- **IsActive** Is the device connected, or has it been seen in the last 14 days?
- **IsPaired** Does the device container require pairing?
- **IsNetworked** Is this a networked device?
- **IsMachineContainer** Is the container the root device itself?
- **FriendlyName** The name of the device container.
- **DiscoveryMethod** The discovery method for the device container.
- **ModelNumber** The model number for the device container.
- **Manufacturer** The manufacturer name for the device container.
- **PartB\_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **objectInstanceid** ContainerId
- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDeviceContainerRemove**

This event indicates that the InventoryDeviceContainer object is no longer present.

The following fields are available:

- **PartB\_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.



### **Microsoft.Windows.Inventory.Core.InventoryDeviceContainerStartSync**

This event indicates that a new set of InventoryDeviceContainerAdd events will be sent.

The following fields are available:

- **PartB\_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

### **Microsoft.Windows.Inventory.Core.InventoryDeviceInterfaceAdd**

This event retrieves information about what sensor interfaces are available on the device.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.
- **Accelerometer3D** Indicates if an Accelerator3D sensor is found.
- **ActivityDetection** Indicates if an Activity Detection sensor is found.
- **AmbientLight** Indicates if an Ambient Light sensor is found.
- **Barometer** Indicates if a Barometer sensor is found.
- **Custom** Indicates if a Custom sensor is found.
- **FloorElevation** Indicates if a Floor Elevation sensor is found.
- **GeomagneticOrientation** Indicates if a Geo Magnetic Orientation sensor is found.
- **GravityVector** Indicates if a Gravity Detector sensor is found.
- **Gyrometer3D** Indicates if a Gyrometer3D sensor is found.
- **Humidity** Indicates if a Humidity sensor is found.
- **LinearAccelerometer** Indicates if a Linear Accelerometer sensor is found.
- **Magnetometer3D** Indicates if a Magnetometer3D sensor is found.
- **Orientation** Indicates if an Orientation sensor is found.
- **Pedometer** Indicates if a Pedometer sensor is found.
- **Proximity** Indicates if a Proximity sensor is found.
- **RelativeOrientation** Indicates if a Relative Orientation sensor is found.
- **SimpleDeviceOrientation** Indicates if a Simple Device Orientation sensor is found.
- **Temperature** Indicates if a Temperature sensor is found.
- **EnergyMeter** Indicates if an Energy sensor is found.

### **Microsoft.Windows.Inventory.Core.InventoryDeviceInterfaceStartSync**

This event indicates that a new set of InventoryDeviceInterfaceAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

### **Microsoft.Windows.Inventory.Core.InventoryDeviceMediaClassAdd**

This event sends additional metadata about a PNP device that is specific to a particular class of devices to help keep Windows up to date while reducing overall size of data payload.

The following fields are available:

- **PartB\_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.
- **Audio\_CaptureDriver** The Audio device capture driver endpoint.
- **Audio\_RenderDriver** The Audio device render driver endpoint.

### **Microsoft.Windows.Inventory.Core.InventoryDeviceMediaClassRemove**

This event indicates that the InventoryDeviceMediaClassRemove object is no longer present.

The following fields are available:

- **PartB\_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDeviceMediaClassStartSync**

This event indicates that a new set of InventoryDeviceMediaClassSAdd events will be sent.

The following fields are available:

- **PartB\_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDevicePnpAdd**

This event sends basic metadata about a PNP device and its associated driver to help keep Windows up-to-date.

The following fields are available:

- **HWID** A JSON array that provides the value and order of the HWID tree for the device.
- **COMPID** A JSON array that provides the value and order of the compatible ID tree for the device.
- **InstallState** The device installation state. One of these values: <https://msdn.microsoft.com/en-us/library/windows/hardware/ff543130.aspx>
- **Enumerator** The bus that enumerated the device.
- **ContainerId** A system-supplied GUID that uniquely groups the functional devices associated with a single-function or multifunction device installed in the device.
- **DeviceState** DeviceState is a bitmask of the following: DEVICE\_IS\_CONNECTED 0x0001 (currently only for container). DEVICE\_IS\_NETWORK\_DEVICE 0x0002 (currently only for container). DEVICE\_IS\_PAIRED 0x0004 (currently only for container). DEVICE\_IS\_ACTIVE 0x0008 (currently never set). DEVICE\_IS\_MACHINE 0x0010 (currently only for container). DEVICE\_IS\_PRESENT 0x0020 (currently always set). DEVICE\_IS\_HIDDEN 0x0040. DEVICE\_IS\_PRINTER 0x0080 (currently only for container). DEVICE\_IS\_WIRELESS 0x0100. DEVICE\_IS\_WIRELESS\_FAT 0x0200. The most common values are therefore: 32 (0x20)= device is present. 96 (0x60)= device is present but hidden. 288 (0x120)= device is a wireless device that is present.
- **ParentId** Device instance id of the parent of the device.
- **STACKID** A JSON array that provides the value and order of the STACKID tree for the device.
- **Description** The device description.
- **MatchingID** Represents the hardware ID or compatible ID that Windows uses to install a device instance.
- **Class** The device setup class of the driver loaded for the device.
- **ClassGuid** The device setup class guid of the driver loaded for the device.
- **Manufacturer** The device manufacturer.
- **Model** The device model.
- **Inf** The INF file name.
- **DriverVerVersion** The version of the driver loaded for the device.
- **DriverVerDate** The date of the driver loaded for the device.
- **Provider** The device provider.
- **DriverPackageStrongName** The immediate parent directory name in the Directory field of InventoryDriverPackage.
- **Service** The device service name.
- **LowerClassFilters** Lower filter class drivers IDs installed for the device.
- **LowerFilters** Lower filter drivers IDs installed for the device.
- **UpperClassFilters** Upper filter class drivers IDs installed for the device.
- **UpperFilters** Upper filter drivers IDs installed for the device.

- **PartB\_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **DriverId** A unique identifier for the installed device.
- **DriverName** The name of the driver image file.
- **InventoryVersion** The version of the inventory file generating the events.
- **ProblemCode** The current error code for the device.

#### **Microsoft.Windows.Inventory.Core.InventoryDevicePnpRemove**

This event indicates that the InventoryDevicePnpRemove object is no longer present.

The following fields are available:

- **PartB\_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDevicePnpStartSync**

This event indicates that a new set of InventoryDevicePnpAdd events will be sent.

The following fields are available:

- **PartB\_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDriverBinaryAdd**

This event sends basic metadata about driver files running on the system to help keep Windows up-to-date.

The following fields are available:

- **DriverName** The file name of the driver.
- **Inf** The name of the INF file.
- **DriverPackageStrongName** The strong name of the driver package.
- **DriverCompany** The company name that developed the driver.
- **DriverChecksum** The checksum of the driver file.
- **DriverTimeStamp** The low 32 bits of the time stamp of the driver file.
- **DriverType** A bitfield of driver attributes: 1. define DRIVER\_MAP\_DRIVER\_TYPE\_PRINTER 0x0001. 2. define DRIVER\_MAP\_DRIVER\_TYPE\_KERNEL 0x0002. 3. define DRIVER\_MAP\_DRIVER\_TYPE\_USER 0x0004. 4. define DRIVER\_MAP\_DRIVER\_IS\_SIGNED 0x0008. 5. define DRIVER\_MAP\_DRIVER\_IS\_INBOX 0x0010. 6. define DRIVER\_MAP\_DRIVER\_IS\_WINQUAL 0x0040. 7. define DRIVER\_MAP\_DRIVER\_IS\_SELF\_SIGNED 0x0020. 8. define DRIVER\_MAP\_DRIVER\_IS\_CI\_SIGNED 0x0080. 9. define DRIVER\_MAP\_DRIVER\_HAS\_BOOT\_SERVICE 0x0100. 10. define DRIVER\_MAP\_DRIVER\_TYPE\_I386 0x10000. 11. define DRIVER\_MAP\_DRIVER\_TYPE\_IA64 0x20000. 12. define DRIVER\_MAP\_DRIVER\_TYPE\_AMD64 0x40000. 13. define DRIVER\_MAP\_DRIVER\_TYPE\_ARM 0x100000. 14. define DRIVER\_MAP\_DRIVER\_TYPE\_THUMB 0x200000. 15. define DRIVER\_MAP\_DRIVER\_TYPE\_ARMNT 0x400000. 16. define DRIVER\_MAP\_DRIVER\_IS\_TIME\_STAMPED 0x800000.
- **DriverInBox** Is the driver included with the operating system?
- **DriverSigned** Is the driver signed?
- **DriverIsKernelMode** Is it a kernel mode driver?
- **DriverVersion** The version of the driver file.
- **ImageSize** The size of the driver file.
- **Product** The product name that is included in the driver file.
- **ProductVersion** The product version that is included in the driver file.
- **WdfVersion** The Windows Driver Framework version.
- **Service** The name of the service that is installed for the device.

- **PartB\_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDriverBinaryRemove**

This event indicates that the InventoryDriverBinary object is no longer present.

The following fields are available:

- **PartB\_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDriverBinaryStartSync**

This event indicates that a new set of InventoryDriverBinaryAdd events will be sent.

The following fields are available:

- **PartB\_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDriverPackageAdd**

This event sends basic metadata about drive packages installed on the system to help keep Windows up-to-date.

The following fields are available:

- **Inf** The INF name of the driver package.
- **ClassGuid** The class GUID for the device driver.
- **Class** The class name for the device driver.
- **Directory** The path to the driver package.
- **Date** The driver package date.
- **Version** The version of the driver package.
- **Provider** The provider for the driver package.
- **SubmissionId** The HLK submission ID for the driver package.
- **PartB\_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.
- **DriverInBox** Is the driver included with the operating system?

#### **Microsoft.Windows.Inventory.Core.InventoryDriverPackageRemove**

This event indicates that the InventoryDriverPackageRemove object is no longer present.

The following fields are available:

- **PartB\_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Core.InventoryDriverPackageStartSync**

This event indicates that a new set of InventoryDriverPackageAdd events will be sent.

The following fields are available:

- **PartB\_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

#### **Microsoft.Windows.Inventory.Indicators.Checksum**

This event summarizes the counts for the InventoryMiscellaneousUexIndicatorAdd events.

The following fields are available:

- **ChecksumDictionary** A count of each operating system indicator.
- **PCFP** Equivalent to the InventoryId field that is found in other core events.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeVBAAdd**

This event provides a summary rollup count of conditions encountered while performing a local scan of Office files, analyzing for known VBA programmability compatibility issues between legacy office version and ProPlus, and between 32 and 64-bit versions

The following fields are available:

- **Design** Count of files with design issues found
- **Design\_x64** Count of files with 64 bit design issues found
- **DuplicateVBA** Count of files with duplicate VBA code
- **HasVBA** Count of files with VBA code
- **Inaccessible** Count of files that were inaccessible for scanning
- **Issues** Count of files with issues detected
- **Issues\_x64** Count of files with 64-bit issues detected
- **IssuesNone** Count of files with no issues detected
- **IssuesNone\_x64** Count of files with no 64-bit issues detected
- **Locked** Count of files that were locked, preventing scanning
- **NoVBA** Count of files with no VBA inside
- **Protected** Count of files that were password protected, preventing scanning
- **RemLimited** Count of files that require limited remediation changes
- **RemLimited\_x64** Count of files that require limited remediation changes for 64-bit issues
- **RemSignificant** Count of files that require significant remediation changes
- **RemSignificant\_x64** Count of files that require significant remediation changes for 64-bit issues
- **Score** Overall compatibility score calculated for scanned content
- **Score\_x64** Overall 64-bit compatibility score calculated for scanned content
- **Total** Total number of files scanned
- **Validation** Count of files that require additional manual validation
- **Validation\_x64** Count of files that require additional manual validation for 64-bit issues

#### **Microsoft.Windows.Inventory.Core.InventoryApplicationFrameworkStartSync**

This event indicates that a new set of InventoryApplicationFrameworkAdd events will be sent

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events

#### **Microsoft.Windows.Inventory.Core.InventoryApplicationFrameworkAdd**

This event provides the basic metadata about the frameworks an application may depend on

The following fields are available:

- **FileId** A hash that uniquely identifies a file
- **Frameworks** The list of frameworks this file depends on
- **InventoryVersion** The version of the inventory file generating the events
- **ProgramId** A hash of the Name, Version, Publisher, and Language of an application used to identify it

#### **Microsoft.Windows.Inventory.Indicators.InventoryMiscellaneousUexIndicatorAdd**

These events represent the basic metadata about the OS indicators installed on the system which are used for keeping the device up-to-date.

The following fields are available:

- **PartB\_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **IndicatorValue** The indicator value

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeVBARuleViolationsStartSync**

This event indicates that a new sync is being generated for this object type.

There are no fields in this event.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeVBAStartSync**

This event indicates that a new sync is being generated for this object type.

There are no fields in this event.

#### **Microsoft.Windows.Inventory.Indicators.InventoryMiscellaneousUexIndicatorRemove**

This event is a counterpart to InventoryMiscellaneousUexIndicatorAdd, indicating that the item has been removed.

There are no additional unique fields in this event.

The following fields are available:

- **PartB\_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.

#### **Microsoft.Windows.Inventory.Indicators.InventoryMiscellaneousUexIndicatorStartSync**

This event indicates that a new set of InventoryMiscellaneousUexIndicatorAdd events will be sent.

The following fields are available:

- **PartB\_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeVBARuleViolationsAdd**

This event provides data on Microsoft Office VBA rule violations, including a rollup count per violation type, giving an indication of remediation requirements for an organization. The event identifier is a unique GUID, associated with the validation rule

The following fields are available:

- **Count** Count of total Microsoft Office VBA rule violations

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeAddInAdd**

This event provides data on the installed Office Add-ins.

- **AddInCLSID** The CLSID key office the Office addin.
- **AddInId** The ID of the Office addin.
- **BinFileTimestamp** The timestamp of the Office addin.
- **BinFileVersion** The version of the Office addin.
- **Description** The description of the Office addin.
- **FileId** The file ID of the Office addin.
- **FriendlyName** The friendly name of the Office addin.
- **FullPath** The full path to the Office addin.
- **LoadBehavior** A Uint32 that describes the load behavior.
- **LoadTime** The load time for the Office addin.
- **OfficeApplication** The Office application for this addin.
- **OfficeArchitecture** The architecture of the addin.
- **OfficeVersion** The Office version for this addin.
- **OutlookCrashingAddin** A boolean value that indicates if crashes have been found for this addin.

- **Provider** The provider name for this addin.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeAddInStartSync**

This event indicates that a new sync is being generated for this object type.

There are no fields in this event.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeIdentifiersAdd**

This event provides data on the installed Office identifiers.

- **OAudienceData** The Office Audience descriptor.
- **OAudienceld** The Office Audience ID.
- **OMID** The Office machine ID.
- **OPlatform** The Office architecture.
- **OVersion** The Office version
- **OTenantId** The Office 365 Tenant GUID.
- **OWowMID** The Office machine ID.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeIdentifiersStartSync**

This event indicates that a new sync is being generated for this object type.

There are no fields in this event.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeIESettingsStartSync**

This event indicates that a new sync is being generated for this object type.

There are no fields in this event.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeProductsStartSync**

This event indicates that a new sync is being generated for this object type.

There are no fields in this event.

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeIESettingsAdd**

This event provides data on the installed Office-related Internet Explorer features.

- **OleFeatureAddon** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OleMachineLockdown** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OleMimeHandling** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OleMimeSniffing** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OleNoAxInstall** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OleNoDownload** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OleObjectCaching** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OlePasswordDisable** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OleSafeBind** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OleSecurityBand** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OleUncSaveCheck** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OleValidateUrl** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OleWebOcPopup** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OleWinRestrict** For more information, see the Office-related [Internet Feature Control Keys](#).
- **OleZoneElevate** For more information, see the Office-related [Internet Feature Control Keys](#).

#### **Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeProductsAdd**

This event describes the Office products that are installed.

- **OC2rApps** The Office Click-to-Run apps.
- **OC2rSkus** The Office Click-to-Run products.
- **OMsiApps** The Office MSI apps.
- **OProductCodes** The Office MSI product code.

## OneDrive events

### **Microsoft.OneDrive.Sync.Setup.APIOperation**

This event includes basic data about install and uninstall OneDrive API operations.

The following fields are available:

- **APIName** The name of the API.
- **ScenarioName** The name of the scenario.
- **Duration** How long the operation took.
- **isSuccess** Was the operation successful?
- **ResultCode** The result code.

### **Microsoft.OneDrive.Sync.Setup.EndExperience**

This event includes a success or failure summary of the installation.

The following fields are available:

- **APIName** The name of the API.
- **ScenarioName** The name of the scenario.
- **Hresult** The HRESULT of the operation.
- **isSuccess** Was the operation successful?

### **Microsoft.OneDrive.Sync.Setup.OSUpgradeInstallationOperation**

This event is related to the OS version when the OS is upgraded with OneDrive installed.

The following fields are available:

- **HResult** The HRESULT of the operation.
- **SourceOSVersion** The source version of the operating system.
- **SourceOSBuildNumber** The source build number of the operating system.
- **SourceOSBuildBranch** The source branch of the operating system.
- **CurrentOSVersion** The current version of the operating system.
- **CurrentOSBuildNumber** The current build number of the operating system.
- **CurrentOSBuildBranch** The current branch of the operating system.
- **CurrentOneDriveVersion** The current version of OneDrive.

### **Microsoft.OneDrive.Sync.Setup.RegisterStandaloneUpdaterAPIOperation**

This event is related to registering or unregistering the OneDrive update task.

The following fields are available:

- **APIName** The name of the API.
- **ScenarioName** The name of the scenario.
- **UnregisterOldTaskResult** The HRESULT of the UnregisterOldTask operation.
- **RegisterNewTaskResult** The HRESULT of the RegisterNewTask operation.
- **isSuccess** Was the operation successful?

### **Microsoft.OneDrive.Sync.Setup.SetupCommonData**



This event contains basic OneDrive configuration data that helps to diagnose failures.

The following fields are available:

- **AppVersion** The version of the app.
- **OfficeVersion** The version of Office that is installed.
- **BuildArch** Is the architecture x86 or x64?
- **Market** Which market is this in?
- **OneDriveDeviceId** The OneDrive device ID.
- **MachineGuid** The CEIP machine ID.
- **IsMSFTInternal** Is this an internal Microsoft device?
- **OSDeviceName** Only if the device is internal to Microsoft, the device name.
- **OSUserName** Only if the device is internal to Microsoft, the user name.
- **Environment** Is the device on the production or int service?
- **OfficeVersionString** The version of Office that is installed.
- **BuildArchitecture** Is the architecture x86 or x64?
- **UserGuid** The CEIP user ID.
- **MSFTInternal** Is this an internal Microsoft device?

#### **Microsoft.OneDrive.Sync.Updater.CommonData**

This event contains basic OneDrive configuration data that helps to diagnose failures.

The following fields are available:

- **AppVersion** The version of the app.
- **OfficeVersion** The version of Office that is installed.
- **BuildArch** Is the architecture x86 or x64?
- **Market** Which market is this in?
- **OneDriveDeviceId** The OneDrive device ID.
- **MachineGuid** The CEIP machine ID.
- **IsMSFTInternal** Is this an internal Microsoft device?
- **OSDeviceName** Only if the device is internal to Microsoft, the device name.
- **OSUserName** Only if the device is internal to Microsoft, the user name.
- **Environment** Is the device on the production or int service?
- **UserGuid** A unique global user identifier.

#### **Microsoft.OneDrive.Sync.Updater.ComponentInstallState**

This event determines the installation state of dependent OneDrive components.

The following fields are available:

- **ComponentName** The name of the dependent component.
- **isInstalled** Is the dependent component installed?

#### **Microsoft.OneDrive.Sync.Updater.OfficeRegistration**

This event determines the status of the OneDrive integration with Microsoft Office.

The following fields are available:

- **isValid** Is the Microsoft Office registration valid?

#### **Microsoft.OneDrive.Sync.Updater.OverlayIconStatus**

This event indicates if the OneDrive overlay icon is working correctly. 0 = healthy; 1 = can be fixed; 2 = broken

The following fields are available:

- **32bit** The status of the OneDrive overlay icon on a 32-bit operating system.
- **64bit** The status of the OneDrive overlay icon on a 64-bit operating system.

#### **Microsoft.OneDrive.Sync.Updater.RepairResult**

The event determines the result of the installation repair.

The following fields are available:

- **hr** The HRESULT of the operation.

#### **Microsoft.OneDrive.Sync.Updater.SetupBinaryDownloadHRESULT**

This event indicates the status when downloading the OneDrive setup file.

The following fields are available:

- **hr** The HRESULT of the operation.

#### **Microsoft.OneDrive.Sync.Updater.UpdateOverallResult**

This event determines the outcome of the operation.

The following fields are available:

- **UpdaterVersion** The version of the updater.
- **IsLoggingEnabled** Is logging enabled?
- **hr** The HRESULT of the operation.

#### **Microsoft.OneDrive.Sync.Updater.UpdateTierReg**

This event determines status of the update tier registry values.

The following fields are available:

- **regReadEnterpriseHr** The HRESULT of the enterprise reg read value.
- **regReadTeamHr** The HRESULT of the team reg read value.

#### **Microsoft.OneDrive.Sync.Updater.UpdateXmlDownloadHRESULT**

This event determines the status when downloading the OneDrive update configuration file.

The following fields are available:

- **hr** The HRESULT of the operation.

#### **Microsoft.OneDrive.Sync.Updater.WebConnectionStatus**

This event determines the error code that was returned when verifying Internet connectivity.

The following fields are available:

- **winInetError** The HRESULT of the operation.

## Setup events

#### **SetupPlatformTel.SetupPlatformTelActivityEvent**

This event sends a unique ID that can be used to bind Setup Platform events together, to help keep Windows up to date.

The following fields are available:

- **FieldName** Retrieves the event name/data point. Examples: InstallStartTime, InstallEndtime, OverallResult etc.

- **GroupName** Retrieves the groupname the event belongs to. Example: Install Information, DU Information, Disk Space Information etc.
- **Value** Retrieves the value associated with the corresponding event name. For example: For time-related events, this will include the system time.
- **ActivityId** Provides a unique Id to correlate events that occur between a activity start event, and a stop event
- **ActivityName** Provides a friendly name of the package type that belongs to the ActivityId (Setup, LanguagePack, GDR, Driver, etc.)

#### **SetupPlatformTel.SetupPlatformTelActivityStarted**

This event sends basic metadata about the update installation process generated by SetupPlatform to help keep Windows up to date.

The following fields are available:

- **Name** The name of the dynamic update type. Example: GDR driver

#### **SetupPlatformTel.SetupPlatformTelActivityStopped**

This event sends basic metadata about the update installation process generated by SetupPlatform to help keep Windows up to date.

#### **SetupPlatformTel.SetupPlatformTelEvent**

This service retrieves events generated by SetupPlatform, the engine that drives the various deployment scenarios.

The following fields are available:

- **FieldName** Retrieves the event name/data point. Examples: InstallStartTime, InstallEndtime, OverallResult etc.
- **Value** Retrieves the value associated with the corresponding event name (Field Name). For example: For time related events this will include the system time.
- **GroupName** Retrieves the groupname the event belongs to. Example: Install Information, DU Information, Disk Space Information etc.

## Shared PC events

#### **Microsoft.Windows.SharedPC.AccountManager.DeleteUserAccount**

Activity for deletion of a user account for devices set up for Shared PC mode as part of the Transient Account Manager to help keep Windows up to date. Deleting unused user accounts on shared devices frees up disk space to improve Windows Update success rates.

The following fields are available:

- **wilActivity** Windows Error Reporting data collected when there is a failure in deleting a user account with the Transient Account Manager.
- **userSid** The security identifier of the account.
- **accountType** The type of account that was deleted. Example: AD, AAD, or Local

#### **Microsoft.Windows.SharedPC.AccountManager.SinglePolicyEvaluation**

Activity for run of the Transient Account Manager that determines if any user accounts should be deleted for devices set up for Shared PC mode to help keep Windows up to date. Deleting unused user accounts on shared devices frees up disk space to improve Windows Update success rates

The following fields are available:

- **wilActivity** Windows Error Reporting data collected when there is a failure in evaluating accounts to be deleted with the Transient Account Manager.
- **totalAccountCount** The number of accounts on a device after running the Transient Account Manager

policies.

- **evaluationTrigger** When was the Transient Account Manager policies ran? Example: At log off or during maintenance hours

## Software update events

### **SoftwareUpdateClientTelemetry.CheckForUpdates**

This event sends tracking data about the software distribution client check for content that is applicable to a device, to help keep Windows up to date

The following fields are available:

- **EventScenario** Indicates the purpose of sending this event - whether because the software distribution just started checking for content, or whether it was cancelled, succeeded, or failed.
- **EventInstanceID** A globally unique identifier for event instance.
- **DeviceModel** What is the device model.
- **BiosName** The name of the device BIOS.
- **BIOSVendor** The vendor of the BIOS.
- **BiosVersion** The version of the BIOS.
- **BiosReleaseDate** The release date of the device BIOS.
- **SystemBIOSMajorRelease** Major version of the BIOS.
- **SystemBIOSMinorRelease** Minor version of the BIOS.
- **BiosFamily** The family of the BIOS (Basic Input Output System).
- **BiosSKUNumber** The sku number of the device BIOS.
- **ClientVersion** The version number of the software distribution client.
- **WUDeviceID** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue.
- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client.
- **ProcessName** The process name of the caller who initiated API calls, in the event where CallerApplicationName was not provided.
- **ServiceGuid** An ID which represents which service the software distribution client is checking for content (Windows Update, Microsoft Store, etc.).
- **StatusCode** Indicates the result of a CheckForUpdates event (success, cancellation, failure code HRESULT).
- **ExtendedStatusCode** Secondary error code for certain scenarios where StatusCode wasn't specific enough.
- **FlightRing** The ring (speed of getting builds) that a device is on if participating in flighting (pre-release builds).
- **FlightBranch** The branch that a device is on if participating in flighting (pre-release builds).
- **RelatedCV** The previous Correlation Vector that was used before swapping with a new one
- **IsWUfBEnabled** Indicates if Windows Update for Business is enabled on the device.
- **IsWUfBDualScanEnabled** Indicates if Windows Update for Business dual scan is enabled on the device.
- **ShippingMobileOperator** The mobile operator that a device shipped on.
- **CurrentMobileOperator** The mobile operator the device is currently connected to.
- **HomeMobileOperator** The mobile operator that the device was originally intended to work with.
- **PhonePreviewEnabled** Indicates whether a phone was getting preview build, prior to flighting (pre-release builds) being introduced.
- **ActivityMatchingId** Contains a unique ID identifying a single CheckForUpdates session from initialization to completion.
- **SyncType** Describes the type of scan the event was
- **IPVersion** Indicates whether the download took place over IPv4 or IPv6

- **NumberOfApplicationsCategoryScanEvaluated** The number of categories (apps) for which an app update scan checked
- **ScanDurationInSeconds** The number of seconds a scan took
- **ScanEnqueueTime** The number of seconds it took to initialize a scan
- **NumberOfLoop** The number of round trips the scan required
- **NumberOfUpdatesEvaluated** The total number of updates which were evaluated as a part of the scan
- **NumberOfNewUpdatesFromServiceSync** The number of updates which were seen for the first time in this scan
- **ServiceUrl** The environment URL a device is configured to scan with
- **Online** Indicates if this was an online scan.
- **AllowCachedResults** Indicates if the scan allowed using cached results.
- **MetadataIntegrityMode** The mode of the update transport metadata integrity check. 0-Unknown, 1-Ignoe, 2-Audit, 3-Enforce
- **TotalNumMetadataSignatures** The total number of metadata signatures checks done for new metadata that was synced down.
- **NumFailedMetadataSignatures** The number of metadata signatures checks which failed for new metadata synced down.
- **MSIError** The last error that was encountered during a scan for updates.
- **DriverError** The error code hit during a driver scan. This is 0 if no error was encountered.
- **FailedUpdatesCount** The number of updates that failed to be evaluated during the scan.
- **FailedUpdateGuids** The GUIDs for the updates that failed to be evaluated during the scan.
- **CapabilityDetectoidGuid** The GUID for a hardware applicability detectoid that could not be evaluated.
- **ExtendedMetadataCabUrl** Hostname that is used to download an update.
- **CDNId** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue.
- **CDNCountryCode** Two letter country abbreviation for the CDN's location.
- **NetworkConnectivityDetected** Indicates the type of network connectivity that was detected. 0 - IPv4, 1 - IPv6
- **NumberOfApplicableUpdates** The number of updates which were ultimately deemed applicable to the system after the detection process is complete
- **ApplicableUpdateInfo** Metadata for the updates which were detected as applicable
- **WebServiceRetryMethods** Web service method requests that needed to be retried to complete operation.
- **DeferredUpdates** Update IDs which are currently being deferred until a later time
- **BranchReadinessLevel** The servicing branch configured on the device.
- **DeferralPolicySources** Sources for any update deferral policies defined (GPO = 0x10, MDM = 0x100, Flight = 0x1000, UX = 0x10000).
- **QualityUpdateDeferral** The deferral period configured for quality OS updates on the device (in days).
- **QualityUpdatePause** Indicates whether quality OS updates are paused on the device.
- **QualityUpdatePausePeriod** The pause duration configured for quality OS updates on the device (in days).
- **FeatureUpdateDeferral** The deferral period configured for feature OS updates on the device (in days).
- **FeatureUpdatePause** Indicates whether feature OS updates are paused on the device.
- **FeatureUpdatePausePeriod** The pause duration configured for feature OS updates on the device (in days).
- **DriverExclusionPolicy** Indicates if the policy for not including drivers with Windows Update is enabled.
- **TargetMetadataVersion** For self-initiated healing, this is the target version of the SIH engine to download (if needed). If not, the value is null.
- **CachedEngineVersion** For self-initiated healing, the version of the SIH engine that is cached on the device. If the SIH engine does not exist, the value is null.
- **SearchFilter** Contains information indicating filters applied while checking for content applicable to the device.

For example, to filter out all content which may require a reboot.

- **IntentPFNs** Intended application-set metadata for atomic update scenarios.
- **PausedUpdates** A list of UpdateIds which that currently being paused.
- **PauseQualityUpdatesStartTime** If quality OS updates are paused on the device, this is the date and time for the beginning of the pause time window.
- **PauseQualityUpdatesEndTime** If quality OS updates are paused on the device, this is the date and time for the end of the pause time window.
- **PauseFeatureUpdatesStartTime** If feature OS updates are paused on the device, this is the date and time for the beginning of the pause time window.
- **PauseFeatureUpdatesEndTime** If feature OS updates are paused on the device, this is the date and time for the end of the pause time window.
- **Context** Gives context on where the error has occurred. Example: AutoEnable, GetSLSData, AddService, Misc, or Unknown
- **DriverSyncPassPerformed** Were drivers scanned this time?

### **SoftwareUpdateClientTelemetry.Commit**

This event sends data on whether the Update Service has been called to execute an upgrade, to help keep Windows up to date.

The following fields are available:

- **EventScenario** State of call
- **EventInstanceID** A globally unique identifier for event instance.
- **DeviceModel** What is the device model.
- **BiosName** The name of the device BIOS.
- **BIOSVendor** The vendor of the BIOS.
- **BiosVersion** The version of the BIOS.
- **BiosReleaseDate** The release date of the device BIOS.
- **SystemBIOSMajorRelease** Major version of the BIOS.
- **SystemBIOSMinorRelease** Minor version of the BIOS.
- **BiosFamily** The family of the BIOS (Basic Input Output System).
- **BiosSKUNumber** The sku number of the device BIOS.
- **ClientVersion** The version number of the software distribution client.
- **WUDeviceID** UniqueDeviceID
- **ServerId** Identifier for the service to which the software distribution client is connecting, such as Windows Update and Microsoft Store.
- **EventType** Possible values are "Child", "Bundle", or "Driver".
- **UpdateId** Unique Update ID
- **BundleId** Identifier associated with the specific content bundle; should not be all zeros if the bundleID was found.
- **RevisionNumber** Unique revision number of Update
- **HandlerType** Indicates the kind of content (app, driver, windows patch, etc.)
- **BundleRevisionNumber** Identifies the revision number of the content bundle
- **FlightId** The specific id of the flight the device is getting
- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client

### **SoftwareUpdateClientTelemetry.Download**

This event sends tracking data about the software distribution client download of the content for that update, to help keep Windows up to date.

The following fields are available:

- **EventScenario** Indicates the purpose of sending this event - whether because the software distribution just started downloading content, or whether it was cancelled, succeeded, or failed.
- **EventInstanceID** A globally unique identifier for event instance.
- **DeviceModel** What is the device model.
- **BiosName** The name of the device BIOS.
- **BIOSVendor** The vendor of the BIOS.
- **BiosVersion** The version of the BIOS.
- **BiosReleaseDate** The release date of the device BIOS.
- **SystemBIOSMajorRelease** Major version of the BIOS.
- **SystemBIOSMinorRelease** Minor version of the BIOS.
- **BiosFamily** The family of the BIOS (Basic Input Output System).
- **BiosSKUNumber** The sku number of the device BIOS.
- **ClientVersion** The version number of the software distribution client.
- **WUDeviceID** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue.
- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client.
- **ProcessName** The process name of the caller who initiated API calls, in the event where CallerApplicationName was not provided.
- **ServiceGuid** An ID which represents which service the software distribution client is installing content for (Windows Update, Microsoft Store, etc.).
- **StatusCode** Indicates the result of a Download event (success, cancellation, failure code HRESULT).
- **ExtendedStatusCode** Secondary error code for certain scenarios where StatusCode wasn't specific enough.
- **FlightRing** The ring (speed of getting builds) that a device is on if participating in flighting (pre-release builds).
- **FlightBranch** The branch that a device is on if participating in flighting (pre-release builds).
- **RelatedCV** The previous Correlation Vector that was used before swapping with a new one
- **IsWUfBEnabled** Indicates if Windows Update for Business is enabled on the device.
- **IsWUfBDualScanEnabled** Indicates if Windows Update for Business dual scan is enabled on the device.
- **ShippingMobileOperator** The mobile operator that a device shipped on.
- **CurrentMobileOperator** The mobile operator the device is currently connected to.
- **HomeMobileOperator** The mobile operator that the device was originally intended to work with.
- **PhonePreviewEnabled** Indicates whether a phone was opted-in to getting preview builds, prior to flighting (pre-release builds) being introduced.
- **IPVersion** Indicates whether the download took place over IPv4 or IPv6.
- **NetworkCostBitMask** Indicates what kind of network the device is connected to (roaming, metered, over data cap, etc.)
- **NetworkRestrictionStatus** More general version of NetworkCostBitMask, specifying whether Windows considered the current network to be "metered."
- **TimeToEstablishConnection** Time (in ms) it took to establish the connection prior to beginning downloaded.
- **HostName** The hostname URL the content is downloading from.
- **CDNId** ID which defines which CDN the software distribution client downloaded the content from.
- **CDNCountryCode** Two letter country abbreviation for the CDN's location.
- **ActiveDownloadTime** How long the download took, in seconds, excluding time where the update wasn't actively being downloaded.
- **IsDependentSet** Indicates whether a driver is a part of a larger System Hardware/Firmware Update
- **TargetingVersion** For drivers targeted to a specific device model, this is the version number of the drivers

being distributed to the device.

- **HardwareId** If this download was for a driver targeted to a particular device model, this ID indicates the model of the device.
- **UpdateImportance** Indicates whether a piece of content was marked as Important, Recommended, or Optional.
- **TargetGroupId** For drivers targeted to a specific device model, this ID indicates the distribution group of devices receiving that driver.
- **RepeatFailFlag** Indicates whether this specific piece of content had previously failed to download.
- **BytesDownloaded** How many bytes were downloaded for an individual piece of content (not the entire bundle).
- **TotalExpectedBytes** The total count of bytes that the download is expected to be.
- **ThrottlingServiceHResult** Result code (success/failure) while contacting a web service to determine whether this device should download content yet.
- **EventType** Possible values are Child, Bundle, or Driver.
- **UpdateId** An identifier associated with the specific piece of content.
- **RevisionNumber** Identifies the revision number of this specific piece of content.
- **BundleId** Identifier associated with the specific content bundle; should not be all zeros if the bundleID was found.
- **BundleRevisionNumber** Identifies the revision number of the content bundle.
- **HandlerType** Indicates what kind of content is being downloaded (app, driver, windows patch, etc.).
- **DownloadPriority** Indicates whether a download happened at background, normal, or foreground priority.
- **FlightId** The specific id of the flight (pre-release build) the device is getting.
- **Setup360Phase** If the download is for an operating system upgrade, this datapoint indicates which phase of the upgrade is underway.
- **UsedDO** Whether the download used the delivery optimization service.
- **CbsDownloadMethod** Indicates whether the download was a full-file download or a partial/delta download.
- **UsedSystemVolume** Indicates whether the content was downloaded to the device's main system storage drive, or an alternate storage drive.
- **FlightBuildNumber** If this download was for a flight (pre-release build), this indicates the build number of that flight.
- **BundleBytesDownloaded** How many bytes were downloaded for the specific content bundle.
- **BundleRepeatFailFlag** Indicates whether this particular update bundle had previously failed to download.
- **DownloadScenarioId** A unique ID for a given download used to tie together WU and DO events.
- **PackageFullName** The package name of the content.
- **AppXBlockHashValidationFailureCount** A count of the number of blocks that have failed validation after being downloaded.
- **CachedEngineVersion** For self-initiated healing, the version of the SIH engine that is cached on the device. If the SIH engine does not exist, the value is null.
- **TargetMetadataVersion** For self-initiated healing, this is the target version of the SIH engine to download (if needed). If not, the value is null.
- **DownloadType** Differentiates the download type of SIH downloads between Metadata and Payload downloads.
- **WUSetting** Indicates the users' current updating settings.
- **ProcessorArchitecture** Processor architecture of the system (x86, AMD64, ARM).
- **PlatformRole** The PowerPlatformRole as defined on MSDN
- **IsAOACDevice** Is it Always On, Always Connected?
- **EventNamespaceId** Indicates whether the event succeeded or failed. Has the format EventType+ Event where Event is Succeeded, Cancelled, Failed, etc.



- **Edition** Indicates the edition of Windows being used.
- **DeviceOEM** What OEM does this device belong to.
- **ClientManagedByWSUSServer** Indicates whether the client is managed by Windows Server Update Services (WSUS).
- **QualityUpdatePause** Indicates whether quality OS updates are paused on the device.
- **FeatureUpdatePause** Indicates whether feature OS updates are paused on the device.
- **AppXDownloadScope** Indicates the scope of the download for application content. For streaming install scenarios, AllContent - non-streaming download, RequiredOnly - streaming download requested content required for launch, AutomaticOnly - streaming download requested automatic streams for the app, and Unknown - for events sent before download scope is determined by the Windows Update client.

### **SoftwareUpdateClientTelemetry.Install**

This event sends tracking data about the software distribution client installation of the content for that update, to help keep Windows up to date.

The following fields are available:

- **EventScenario** Indicates the purpose of sending this event - whether because the software distribution just started installing content, or whether it was cancelled, succeeded, or failed.
- **EventInstanceID** A globally unique identifier for event instance.
- **DeviceModel** What is the device model.
- **BiosName** The name of the device BIOS.
- **BIOSVendor** The vendor of the BIOS.
- **BiosVersion** The version of the BIOS.
- **BiosReleaseDate** The release date of the device BIOS.
- **SystemBIOSMajorRelease** Major version of the BIOS.
- **SystemBIOSMinorRelease** Minor version of the BIOS.
- **BiosFamily** The family of the BIOS (Basic Input Output System).
- **BiosSKUNumber** The sku number of the device BIOS.
- **ClientVersion** The version number of the software distribution client.
- **WUDeviceID** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue.
- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client.
- **ProcessName** The process name of the caller who initiated API calls, in the event where CallerApplicationName was not provided.
- **ServiceGuid** An ID which represents which service the software distribution client is installing content for (Windows Update, Microsoft Store, etc.).
- **StatusCode** Indicates the result of an installation event (success, cancellation, failure code HRESULT).
- **ExtendedStatusCode** Secondary error code for certain scenarios where StatusCode wasn't specific enough.
- **FlightRing** The ring that a device is on if participating in the Windows Insider Program.
- **FlightBranch** The branch that a device is on if participating in the Windows Insider Program.
- **RelatedCV** The previous Correlation Vector that was used before swapping with a new one
- **IsWUfBEnabled** Is Windows Update for Business enabled on the device?
- **IsWUfBDualScanEnabled** Is Windows Update for Business dual scan enabled on the device?
- **ShippingMobileOperator** The mobile operator that a device shipped on.
- **CurrentMobileOperator** Mobile operator that device is currently connected to.
- **HomeMobileOperator** The mobile operator that the device was originally intended to work with.
- **PhonePreviewEnabled** Indicates whether a phone was getting preview build, prior to flighting being

introduced.

- **TargetGroupID** For drivers targeted to a specific device model, this ID indicates the distribution group of devices receiving that driver.
- **RepeatFailFlag** Indicates whether this specific piece of content had previously failed to install.
- **EventType** Possible values are Child, Bundle, or Driver.
- **TargetingVersion** For drivers targeted to a specific device model, this is the version number of the drivers being distributed to the device.
- **UpdateImportance** Indicates whether a piece of content was marked as Important, Recommended, or Optional.
- **IsFirmware** Is this update a firmware update?
- **IsFinalOutcomeEvent** Does this event signal the end of the update/upgrade process?
- **IsDependentSet** Is the driver part of a larger System Hardware/Firmware update?
- **DriverPingBack** Contains information about the previous driver and system state.
- **ExtendedErrorCode** The extended error code.
- **CSLErrorType** The stage of CBS installation where it failed.
- **MsiAction** The stage of MSI installation where it failed.
- **MsiProductCode** The unique identifier of the MSI installer.
- **TransactionCode** The ID which represents a given MSI installation
- **HardwareId** If this install was for a driver targeted to a particular device model, this ID indicates the model of the device.
- **IsSuccessFailurePostReboot** Did it succeed and then fail after a restart?
- **UpdateId** Unique update ID
- **RevisionNumber** The revision number of this specific piece of content.
- **BundleId** Identifier associated with the specific content bundle; should not be all zeros if the bundleID was found.
- **BundleRevisionNumber** Identifies the revision number of the content bundle.
- **HandlerType** Indicates what kind of content is being installed. Example: app, driver, Windows update
- **FlightId** The specific ID of the Windows Insider build the device is getting.
- **Setup360Phase** If the install is for an operating system upgrade, indicates which phase of the upgrade is underway.
- **UsedSystemVolume** Indicates whether the content was downloaded and then installed from the device's main system storage drive, or an alternate storage drive.
- **FlightBuildNumber** If this installation was for a Windows Insider build, this is the build number of that build.
- **BundleRepeatFailFlag** Has this particular update bundle previously failed to install?
- **PackageFullName** The package name of the content being installed.
- **CachedEngineVersion** For self-initiated healing, the version of the SIH engine that is cached on the device. If the SIH engine does not exist, the value is null.
- **BundleBytesDownloaded** How many bytes were downloaded for the specific content bundle?
- **CbsDownloadMethod** Was the download a full download or a partial download?
- **ClientManagedByWSUSServer** Is the client managed by Windows Server Update Services (WSUS)?
- **DeviceOEM** What OEM does this device belong to.
- **DownloadPriority** The priority of the download activity.
- **DownloadScenarioId** A unique ID for a given download used to tie together WU and DO events.
- **Edition** Indicates the edition of Windows being used.
- **EventNamespaceID** Indicates whether the event succeeded or failed. Has the format EventType+Event where Event is Succeeded, Cancelled, Failed, etc.
- **IsAOACDevice** Is it Always On, Always Connected? (Mobile device usage model)

- **PlatformRole** The PowerPlatformRole as defined on MSDN.
- **ProcessorArchitecture** Processor architecture of the system (x86, AMD64, ARM).
- **RepeatSuccessInstallFlag** Indicates whether this specific piece of content had previously installed successfully, for example if another user had already installed it.
- **WUSetting** Indicates the user's current updating settings.
- **IntentPFNs** Intended application-set metadata for atomic update scenarios.
- **QualityUpdatePause** Are quality OS updates paused on the device?
- **FeatureUpdatePause** Are feature OS updates paused on the device?
- **MergedUpdate** Was the OS update and a BSP update merged for installation?

#### **SoftwareUpdateClientTelemetry.SLSDiscovery**

This event sends data about the ability of Windows to discover the location of a backend server with which it must connect to perform updates or content acquisition, in order to determine disruptions in availability of update services and provide context for Windows Update errors.

The following fields are available:

- **EventScenario** Indicates the purpose of sending this event - whether because the software distribution just started checking for content, or whether it was cancelled, succeeded, or failed
- **SusClientId** The unique device ID controlled by the software distribution client
- **WUAVersion** The version number of the software distribution client
- **ServiceID** An ID which represents which service the software distribution client is connecting to (Windows Update, Microsoft Store, etc.)
- **UriPath** Path to the SLS cab that was downloaded
- **HResult** Indicates the result code of the event (success, cancellation, failure code HRESULT)
- **IsBackground** Indicates whether the SLS discovery event took place in the foreground or background
- **NextExpirationTime** Indicates when the SLS cab expires

#### **SoftwareUpdateClientTelemetry.UpdateDetected**

This event sends data about an AppX app that has been updated from the Microsoft Store, including what app needs an update and what version/architecture is required, in order to understand and address problems with apps getting required updates.

The following fields are available:

- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client
- **ApplicableUpdateInfo** Metadata for the updates which were detected as applicable
- **NumberOfApplicableUpdates** The number of updates which were ultimately deemed applicable to the system after the detection process is complete
- **WUDeviceID** The unique device ID controlled by the software distribution client
- **RelatedCV** The previous Correlation Vector that was used before swapping with a new one
- **EventScenario** Indicates the purpose of sending this event - whether because the software distribution just started checking for content, or whether it was cancelled, succeeded, or failed
- **EventInstanceId** A globally unique identifier for event instance
- **DeviceModel** The device's model as defined in system bios
- **BiosName** The name of the device's system bios
- **BIOSVendor** The vendor of the device's system bios
- **BiosVersion** The version of the device's system bios
- **BiosReleaseDate** The release date of the device's system bios
- **SystemBIOSMajorRelease** The major release version of the device's system bios

- **SystemBIOSMinorRelease** The minor release version of the device's system bios
- **BiosFamily** The device's family as defined in system bios
- **BiosSKUNumber** The device's SKU as defined in system bios
- **ClientVersion** The version number of the software distribution client
- **ProcessName** The process name of the caller who initiated API calls, in the event where CallerApplicationName was not provided
- **ServiceGuid** An ID which represents which service the software distribution client is connecting to (Windows Update, Microsoft Store, etc.)
- **StatusCode** Indicates the result code of the event (success, cancellation, failure code HRESULT)
- **ExtendedStatusCode** Secondary status code for certain scenarios where StatusCode wasn't specific enough
- **FlightRing** The ring (speed of getting builds) that a device is on if participating in flighting (pre-release builds).
- **FlightBranch** The branch that a device is on if participating in flighting (pre-release builds).
- **ShippingMobileOperator** The mobile operator that a device shipped on.
- **CurrentMobileOperator** The mobile operator the device is currently connected to.
- **HomeMobileOperator** The mobile operator that the device was originally intended to work with
- **PhonePreviewEnabled** Indicates whether a phone was getting preview build, prior to flighting (pre-release builds) being introduced.
- **ActivityMatchingId** Contains a unique ID identifying a single CheckForUpdates session from initialization to completion
- **SyncType** Describes the type of scan the event was
- **IPVersion** Indicates whether the download took place over IPv4 or IPv6
- **NumberOfApplicationsCategoryScanEvaluated** The number of categories (apps) for which an app update scan checked
- **ScanDurationInSeconds** The number of seconds a scan took
- **ScanEnqueueTime** The number of seconds it took to initialize a scan
- **NumberOfLoop** The number of round trips the scan required
- **NumberOfUpdatesEvaluated** The total number of updates which were evaluated as a part of the scan
- **NumberOfNewUpdatesFromServiceSync** The number of updates which were seen for the first time in this scan
- **ServiceUrl** The environment URL a device is configured to scan with
- **IntentPFNs** Intended application-set metadata for atomic update scenarios.

### SoftwareUpdateClientTelemetry.UpdateMetadataIntegrity

This event identifies whether updates have been tampered with and protects against man-in-the-middle attacks.

The following fields are available:

- **EventScenario** The purpose of this event, such as scan started, scan succeeded, or scan failed.
- **ServiceGuid** Identifies the service to which the software distribution client is connected, Example: Windows Update or Microsoft Store
- **MetadataIntegrityMode** The mode of the transport metadata integrity check. 0 = unknown; 1 = ignore; 2 = audit; 3 = enforce
- **StatusCode** The status code of the event.
- **ExtendedStatusCode** The secondary status code of the event.
- **RevisionId** The revision ID for a specific piece of content.
- **UpdateId** The update ID for a specific piece of content.
- **RevisionNumber** The revision number for a specific piece of content.
- **TimestampTokenId** The time this was created. It is encoded in a timestamp blob and will be zero if the token is malformed.

- **LeafCertId** Integral ID from the FragmentSigning data for certificate that failed.
- **SHA256OfLeafCertPublicKey** A base64 encoding of the hash of the Base64CertData in the FragmentSigning data of the leaf certificate.
- **MetadataSignature** A base64-encoded string of the signature associated with the update metadata (specified by revision ID).
- **SignatureAlgorithm** The hash algorithm for the metadata signature.
- **SHA256OfTimestampToken** A base64-encoded string of hash of the timestamp token blob.
- **ValidityWindowInDays** The validity window that's in effect when verifying the timestamp.
- **TimestampTokenCertThumbprint** The thumbprint of the encoded timestamp token.
- **RawMode** The raw unparsed mode string from the SLS response. This field is null if not applicable.
- **RawValidityWindowInDays** The raw unparsed validity window string in days of the timestamp token. This field is null if not applicable.
- **SHA256OfLeafCerData** A base64 encoding of the hash for the Base64CerData in the FragmentSigning data of the leaf certificate.
- **ListOfSHA256OfIntermediateCerData** A semicolon delimited list of base64 encoding of hashes for the Base64CerData in the FragmentSigning data of an intermediate certificate.
- **EndpointUrl** The endpoint URL where the device obtains update metadata. This is used to distinguish between test, staging, and production environments.
- **SLSPrograms** A test program to which a device may have opted in. Example: Insider Fast

## Update events

### Update360Telemetry.UpdateAgent\_DownloadRequest

This event sends data during the download request phase of updating Windows.

The following fields are available:

- **ErrorCode** The error code returned for the current download request phase.
- **PackageCountTotal** Total number of packages needed.
- **PackageCountRequired** Number of required packages requested.
- **PackageCountOptional** Number of optional packages requested.
- **ObjectId** Unique value for each Update Agent mode.
- **SessionId** Unique value for each Update Agent mode attempt.
- **ScenarioId** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **RelatedCV** Correlation vector value generated from the latest USO scan.
- **Result** Result of the download request phase of update.
- **PackageSizeCanonical** Size of canonical packages in bytes
- **PackageSizeDiff** Size of diff packages in bytes
- **PackageSizeExpress** Size of express packages in bytes
- **FlightId** Unique ID for each flight.
- **UpdateId** Unique ID for each update.
- **PackageCountTotalCanonical** Total number of canonical packages.
- **PackageCountTotalDiff** Total number of diff packages.
- **PackageCountTotalExpress** Total number of express packages.
- **RangeRequestState** Represents the state of the download range request.
- **DeletedCorruptFiles** Indicates if UpdateAgent found any corrupt payload files and whether the payload was deleted.

### Update360Telemetry.UpdateAgent\_Initialize

This event sends data during the initialize phase of updating Windows.

The following fields are available:

- **ErrorCode** The error code returned for the current initialize phase.
- **SessionData** Contains instructions to update agent for processing FODs and DUICs (Null for other scenarios).
- **UpdateId** Unique ID for each update.
- **FlightId** Unique ID for each flight.
- **FlightMetadata** Contains the FlightId and the build being flighted.
- **ObjectId** Unique value for each Update Agent mode.
- **SessionId** Unique value for each Update Agent mode attempt .
- **ScenarioId** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **RelatedCV** Correlation vector value generated from the latest USO scan.
- **Result** Result of the initialize phase of update. 0 = Succeeded, 1 = Failed, 2 = Cancelled, 3 = Blocked, 4 = BlockCancelled

### **Update360Telemetry.UpdateAgent\_Install**

This event sends data during the install phase of updating Windows.

The following fields are available:

- **ErrorCode** The error code returned for the current install phase.
- **ObjectId** Unique value for each Update Agent mode.
- **SessionId** Unique value for each Update Agent mode attempt.
- **ScenarioId** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **RelatedCV** Correlation vector value generated from the latest scan.
- **Result** Result of the install phase of update. 0 = Succeeded 1 = Failed, 2 = Cancelled, 3 = Blocked, 4 = BlockCancelled
- **FlightId** Unique ID for each flight.
- **UpdateId** Unique ID for each update.

### **Update360Telemetry.UpdateAgent\_ModeStart**

This event sends data for the start of each mode during the process of updating Windows.

The following fields are available:

- **Mode** Indicates that the Update Agent mode that has started. 1 = Initialize, 2 = DownloadRequest, 3 = Install, 4 = Commit
- **ObjectId** Unique value for each Update Agent mode.
- **SessionId** Unique value for each Update Agent mode attempt.
- **ScenarioId** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **RelatedCV** The correlation vector value generated from the latest scan.
- **FlightId** Unique ID for each flight.
- **UpdateId** Unique ID for each update.

### **Update360Telemetry.UpdateAgent\_SetupBoxLaunch**

This event sends data during the launching of the setup box when updating Windows.

The following fields are available:

- **Quiet** Indicates whether setup is running in quiet mode. 0 = false 1 = true

- **ObjectId** Unique value for each Update Agent mode.
- **SessionId** Unique value for each Update Agent mode attempt.
- **ScenarioId** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **RelatedCV** Correlation vector value generated from the latest scan.
- **FlightId** Unique ID for each flight.
- **UpdateId** Unique ID for each update.
- **SetupMode** Setup mode 1 = predownload, 2 = install, 3 = finalize
- **SandboxSize** The size of the sandbox folder on the device.

## Upgrade events

### Setup360Telemetry.Downlevel

This event sends data indicating that the device has invoked the downlevel phase of the upgrade. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **ClientId** If using Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, the default value is Media360, but it can be overwritten by the caller to a unique value.
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe.
- **ReportId** In the Windows Update scenario, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Wuid** This is the Windows Update Client ID. In the Windows Update scenario, this is the same as the clientId.
- **TestId** A string that uniquely identifies a group of events.
- **State** Exit state of given Setup360 run. Example: succeeded, failed, blocked, cancelled
- **HostOsSkuName** The operating system edition which is running Setup360 instance (downlevel OS).
- **HostOSBuildNumber** The build number of the downlevel OS.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. It's an HRESULT error code that can be used to diagnose errors.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **SetupVersionBuildNumber** The build number of Setup360 (build number of the target OS).

### Setup360Telemetry.Finalize

This event sends data indicating that the device has invoked the finalize phase of the upgrade, to help keep Windows up-to-date.

The following fields are available:

- **ClientId** With Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe
- **ReportId** With Windows Update, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Wuid** This is the Windows Update Client ID. With Windows Update, this is the same as the clientId.
- **TestId** A string to uniquely identify a group of events.
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **HostOsSkuName** The OS edition which is running Setup360 instance (previous OS).
- **HostOSBuildNumber** The build number of the previous OS.

- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that is used to diagnose errors.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).

### Setup360Telemetry.OsUninstall

The event sends data regarding OS updates and upgrades from Windows 7, Windows 8, and Windows 10. Specifically, the Setup360Telemetry.OSUninstall indicates the outcome of an OS uninstall.

The following fields are available:

- **ClientId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe.
- **ReportId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, this is the GUID for the install.wim.
- **Wuid** Windows Update client ID.
- **TestId** A string to uniquely identify a group of events.
- **State** Exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **HostOsSkuName** The OS edition which is running the Setup360 instance (previous OS).
- **HostOSBuildNumber** The build number of the previous OS.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that is used to diagnose errors.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).

### Setup360Telemetry.PostRebootInstall

This event sends data indicating that the device has invoked the postrebootinstall phase of the upgrade, to help keep Windows up-to-date.

The following fields are available:

- **ClientId** With Windows Update, this is the Windows Update client ID that is passed to Setup. In Media setup, the default value is Media360, but can be overwritten by the caller to a unique value.
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe.
- **ReportId** With Windows Update, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Wuid** This is the Windows Update Client ID. With Windows Update, this is the same as ClientId.
- **TestId** A string to uniquely identify a group of events.
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **HostOsSkuName** The OS edition which is running Setup360 instance (previous OS).
- **HostOSBuildNumber** The build number of the previous OS.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that's used to diagnose errors.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened



- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).

#### Setup360Telemetry.PreDownloadQuiet

This event sends data indicating that the device has invoked the predownload quiet phase of the upgrade, to help keep Windows up to date.

The following fields are available:

- **ClientId** Using Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe
- **ReportId** Using Windows Update, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Wuid** This is the Windows Update Client ID. Using Windows Update, this is the same as the clientId.
- **TestId** A string to uniquely identify a group of events.
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, canceled
- **HostOsSkuName** The OS edition which is running Setup360 instance (previous operating system).
- **HostOSBuildNumber** The build number of the previous OS.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that is used to diagnose errors.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).

#### Setup360Telemetry.PreDownloadUX

The event sends data regarding OS updates and upgrades from Windows 7, Windows 8, and Windows 10. Specifically, the Setup360Telemetry.PredownloadUX indicates the outcome of the PredownloadUX portion of the update process.

The following fields are available:

- **ClientId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **InstanceId** Unique GUID that identifies each instance of setuphost.exe.
- **ReportId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, this is the GUID for the install.wim.
- **Wuid** Windows Update client ID.
- **TestId** A string to uniquely identify a group of events.
- **State** The exit state of the Setup360 run. Example: succeeded, failed, blocked, cancelled
- **HostOsSkuName** The OS edition which is running the Setup360 instance (previous operating system).
- **HostOSBuildNumber** The build number of the previous operating system.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that can be used to diagnose errors.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **SetupVersionBuildNumber** The build number of Setup360 (build number of the target OS).

#### Setup360Telemetry.PreInstallQuiet

This event sends data indicating that the device has invoked the preinstall quiet phase of the upgrade, to help keep Windows up to date.

The following fields are available:

- **ClientId** With Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe
- **ReportId** With Windows Update, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Wuld** This is the Windows Update Client ID. With Windows Update, this is the same as the clientId.
- **TestId** A string to uniquely identify a group of events.
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **HostOsSkuName** The OS edition which is running Setup360 instance (previous OS).
- **HostOSBuildNumber** The build number of the previous OS.
- **Setup360Scenario** Setup360 flow type (Boot, Media, Update, MCT)
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback etc.
- **Setup360Result** The result of Setup360. This is an HRESULT error code that can be used to diagnose errors.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).

#### Setup360Telemetry.PreInstallUX

This event sends data regarding OS updates and upgrades from Windows 7, Windows 8, and Windows 10. Specifically, the Setup360Telemetry.PreinstallUX indicates the outcome of the PreinstallUX portion of the update process.

The following fields are available:

- **ClientId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe.
- **ReportId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, this is the GUID for the install.wim.
- **Wuld** Windows Update client ID.
- **TestId** A string to uniquely identify a group of events.
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **HostOsSkuName** The OS edition which is running the Setup360 instance (previous OS).
- **HostOSBuildNumber** The build number of the previous OS.
- **Setup360Scenario** The Setup360 flow type, Example: Boot, Media, Update, MCT
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that is used to diagnose errors.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).

#### Setup360Telemetry.Setup360

This event sends data about OS deployment scenarios, to help keep Windows up-to-date.

The following fields are available:

- **InstanceId** Retrieves a unique identifier for each instance of a setup session.
- **ReportId** Retrieves the report ID.
- **FlightData** Specifies a unique identifier for each group of Windows Insider builds.

- **ScenarioId** Retrieves the deployment scenario.
- **FieldName** Retrieves the data point.
- **Value** Retrieves the value associated with the corresponding FieldName.
- **ClientId** Retrieves the upgrade ID: Upgrades via Windows Update - specifies the WU clientId. All other deployment - static string.

### Setup360Telemetry.UnexpectedEvent

This event sends data indicating that the device has invoked the unexpected event phase of the upgrade, to help keep Windows up to date.

The following fields are available:

- **ClientId** With Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe
- **ReportId** With Windows Update, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Wuld** This is the Windows Update Client ID. With Windows Update, this is the same as the clientId.
- **TestId** A string to uniquely identify a group of events.
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **HostOsSKUName** The OS edition which is running Setup360 instance (previous OS).
- **HostOSBuildNumber** The build number of the previous OS.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that can be used to diagnose errors.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).

## Windows Error Reporting events

### Microsoft.Windows.WERVertical.OSCrash

This event sends binary data from the collected dump file whenever a bug check occurs, to help keep Windows up to date. This is the OneCore version of this event.

The following fields are available:

- **ReportId** WER Report Id associated with this bug check (used for finding the corresponding report archive in Watson).
- **BugCheckCode** UInt64 "bugcheck code" that identifies a proximate cause of the bug check.
- **BugCheckParameter1** UInt64 parameter providing additional information.
- **BootId** UInt32 identifying the boot number for this device.
- **BugCheckParameter2** UInt64 parameter providing additional information.
- **BugCheckParameter4** UInt64 parameter providing additional information.
- **BugCheckParameter3** UInt64 parameter providing additional information.
- **IsValidDumpFile** True if the dump file is valid for the debugger, false otherwise
- **DumpFileSize** Size of the dump file
- **DumpFileAttributes** Codes that identify the type of data contained in the dump file

## Microsoft Store events

### **Microsoft.Windows.StoreAgent.Telemetry.AbortedInstallation**

This event is sent when an installation or update is canceled by a user or the system and is used to help keep Windows Apps up to date and secure.

The following fields are available:

- **PFN** The product family name of the product being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **HResult** The result code of the last action performed before this operation.
- **IsUpdate** Flag indicating if this is an update.
- **AttemptNumber** Number of retry attempts before it was canceled.
- **CategoryId** The Item Category ID.
- **ProductId** The identity of the package or packages being installed.
- **IsInteractive** Was this requested by a user?
- **IsRemediation** Was this a remediation install?
- **BundleId** The Item Bundle ID.
- **IsMandatory** Was this a mandatory update?
- **SystemAttemptNumber** The total number of automatic attempts at installation before it was canceled.
- **UserAttemptNumber** The total number of user attempts at installation before it was canceled.
- **IsRestore** Is this automatically restoring a previously acquired product?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **IsBundle** Is this a bundle?
- **WUContentId** The Windows Update content ID
- **AggregatedPackageFullNames** The names of all packages to be downloaded and installed.

### **Microsoft.Windows.StoreAgent.Telemetry.BeginGetInstalledContentIds**

This event is sent when an inventory of the apps installed is started to determine whether updates for those apps are available. It's used to help keep Windows up-to-date and secure.

### **Microsoft.Windows.StoreAgent.Telemetry.BeginUpdateMetadataPrepare**

This event is sent when the Store Agent cache is refreshed with any available package updates. It's used to help keep Windows up-to-date and secure.

### **Microsoft.Windows.StoreAgent.Telemetry.CancelInstallation**

This event is sent when an app update or installation is canceled while in interactive mode. This can be canceled by the user or the system. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **IsInteractive** Was this requested by a user?
- **AttemptNumber** Total number of installation attempts.
- **BundleId** The identity of the Windows Insider build that is associated with this product.
- **PreviousHResult** The previous HResult code.
- **ClientAppId** The identity of the app that initiated this operation.
- **CategoryId** The identity of the package or packages being installed.
- **PFN** The name of all packages to be downloaded and installed.
- **ProductId** The name of the package or packages requested for installation.
- **IsUpdate** Is this a product update?
- **IsRemediation** Is this repairing a previous installation?
- **RelatedCV** Correlation Vector of a previous performed action on this product.
- **PreviousInstallState** Previous installation state before it was canceled.

- **IsMandatory** Is this a mandatory update?
- **SystemAttemptNumber** Total number of automatic attempts to install before it was canceled.
- **UserAttemptNumber** Total number of user attempts to install before it was canceled.
- **IsRestore** Is this an automatic restore of a previously acquired product?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **IsBundle** Is this a bundle?
- **WUContentId** The Windows Update content ID
- **AggregatedPackageFullNames** The names of all package or packages to be downloaded and installed.

#### **Microsoft.Windows.StoreAgent.Telemetry.CompleteInstallOperationRequest**

This event is sent after the app installations or updates. It's used to help keep Windows up-to-date and secure

The following fields are available:

- **IsBundle** Is this a bundle?
- **ProductId** The Store Product ID of the product being installed.
- **Skuld** Specific edition of the item being installed.
- **CatalogId** The Store Product ID of the app being installed.
- **PackageFamilyName** The name of the package being installed.
- **HResult** HRESULT code of the action being performed.

#### **Microsoft.Windows.StoreAgent.Telemetry.EndAcquireLicense**

This event is sent after the license is acquired when a product is being installed. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **PFN** Product Family Name of the product being installed.
- **HResult** HRESULT code to show the result of the operation (success/failure).
- **ProductId** The Store Product ID for the product being installed.
- **IsInteractive** Did the user initiate the installation?
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **IsRemediation** Is this repairing a previous installation?
- **UpdateId** The update ID (if this is an update)
- **AttemptNumber** The total number of attempts to acquire this product.
- **IsUpdate** Is this an update?
- **IsMandatory** Is this a mandatory update?
- **SystemAttemptNumber** The number of attempts by the system to acquire this product.
- **UserAttemptNumber** The number of attempts by the user to acquire this product
- **IsRestore** Is this happening after a device restore?
- **IsBundle** Is this a bundle?
- **WUContentId** The Windows Update content ID
- **ParentBundleId** The product's parent bundle ID.
- **AggregatedPackageFullNames** Includes a set of package full names for each app that is part of an atomic set.

#### **Microsoft.Windows.StoreAgent.Telemetry.EndDownload**

This event happens during the app update or installation when content is being downloaded at the end of the process to report success or failure. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **PFN** The Product Family Name of the app being download.
- **IsRemediation** Is this repairing a previous installation?
- **DownloadSize** The total size of the download.
- **ClientAppId** The identity of the app that initiated this operation.
- **CategoryId** The identity of the package or packages being installed.
- **IsUpdate** Is this an update?
- **HResult** The result code of the last action performed.
- **IsInteractive** Is this initiated by the user?
- **AttemptNumber** Number of retry attempts before it was canceled.
- **BundleId** The identity of the Windows Insider build associated with this product.
- **ProductId** The Store Product ID for the product being installed.
- **IsMandatory** Is this a mandatory installation?
- **SystemAttemptNumber** The number of attempts by the system to download.
- **UserAttemptNumber** The number of attempts by the user to download.
- **IsRestore** Is this a restore of a previously acquired product?
- **ParentBundleId** The parent bundle ID (if it's part of a bundle).
- **IsBundle** Is this a bundle?
- **WUContentId** The Windows Update content ID.
- **ExtendedHResult** Any extended HRESULT error codes.
- **AggregatedPackageFullNames** The name of all packages to be downloaded and installed.

#### **Microsoft.Windows.StoreAgent.Telemetry.EndFrameworkUpdate**

This event happens when an app update requires an updated Framework package and the process starts to download it. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **HResult** The result code of the last action performed before this operation.

#### **Microsoft.Windows.StoreAgent.Telemetry.EndGetInstalledContentIds**

This event is sent after sending the inventory of the products installed to determine whether updates for those products are available. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **HResult** The result code of the last action performed before this operation.

#### **Microsoft.Windows.StoreAgent.Telemetry.EndInstall**

This event is sent after a product has been installed. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **BundleId** The identity of the build associated with this product.
- **PFN** Product Family Name of the product being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **CategoryId** The identity of the package or packages being installed.
- **ProductId** The Store Product ID for the product being installed.
- **AttemptNumber** The number of retry attempts before it was canceled.
- **HResult** The result code of the last action performed.
- **IsRemediation** Is this repairing a previous installation?

- **IsInteractive** Is this an interactive installation?
- **IsUpdate** Is this an update?
- **IsMandatory** Is this a mandatory installation?
- **SystemAttemptNumber** The total number of system attempts.
- **UserAttemptNumber** The total number of user attempts.
- **IsRestore** Is this automatically restoring a previously acquired product?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **IsBundle** Is this a bundle?
- **WUContentId** The Windows Update content ID
- **ExtendedHResult** The extended HRESULT error code.
- **AggregatedPackageFullNames** The names of all packages to be downloaded and installed.

#### **Microsoft.Windows.StoreAgent.Telemetry.EndScanForUpdates**

This event is sent after a scan for product updates to determine if there are packages to install. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **HResult** The result code of the last action performed.
- **IsApplicability** Is this request to only check if there are any applicable packages to install?
- **IsInteractive** Is this user requested?
- **ClientAppId** The identity of the app that initiated this operation.
- **IsOnline** Is the request doing an online check?

#### **Microsoft.Windows.StoreAgent.Telemetry.EndSearchUpdatePackages**

This event is sent after searching for update packages to install. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **IsRemediation** Is this repairing a previous installation?
- **IsUpdate** Is this an update?
- **ClientAppId** The identity of the app that initiated this operation.
- **HResult** The result code of the last action performed.
- **ProductId** The Store Product ID for the product being installed.
- **AttemptNumber** The total number of retry attempts before it was canceled.
- **IsInteractive** Is this user requested?
- **PFN** The name of the package or packages requested for install.
- **BundleId** The identity of the build associated with this product.
- **CategoryId** The identity of the package or packages being installed.
- **IsMandatory** Is this a mandatory update?
- **SystemAttemptNumber** The total number of system attempts.
- **UserAttemptNumber** The total number of user attempts.
- **IsRestore** Is this restoring previously acquired content?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **IsBundle** Is this a bundle?
- **WUContentId** The Windows Update content ID
- **AggregatedPackageFullNames** The names of all packages to be downloaded and installed.

#### **Microsoft.Windows.StoreAgent.Telemetry.EndStageUserData**

This event is sent between download and installation to see if there is app data that needs to be restored from the

cloud. It's used to keep Windows up-to-date and secure.

The following fields are available:

- **IsInteractive** Is this user requested?
- **PFN** The name of the package or packages requested for install.
- **IsUpdate** Is this an update?
- **CategoryId** The identity of the package or packages being installed.
- **HResult** The result code of the last action performed.
- **AttemptNumber** The total number of retry attempts before it was canceled.
- **ProductId** The Store Product ID for the product being installed.
- **BundleId** The identity of the build associated with this product.
- **IsRemediation** Is this repairing a previous installation?
- **ClientAppId** The identity of the app that initiated this operation.
- **IsMandatory** Is this a mandatory update?
- **SystemAttemptNumber** The total number of system attempts.
- **UserAttemptNumber** The total number of system attempts.
- **IsRestore** Is this restoring previously acquired content?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **IsBundle** Is this a bundle?
- **WUContentId** The Windows Update content ID
- **AggregatedPackageFullNames** The name of all packages to be downloaded and installed.

#### **Microsoft.Windows.StoreAgent.Telemetry.EndUpdateMetadataPrepare**

This event happens after a scan for available app updates. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **HResult** The result code of the last action performed.

#### **Microsoft.Windows.StoreAgent.Telemetry.FulfillmentComplete**

This event is sent at the end of an app install or update and is used to track the very end of the install or update process.

The following fields are available:

- **ProductId** The product ID of the app that is being updated or installed.
- **PFN** The Package Family Name of the app that is being installed or updated.
- **FailedRetry** Was the installation or update retry successful?
- **HResult** The HRESULT code of the operation.

#### **Microsoft.Windows.StoreAgent.Telemetry.FulfillmentInitiate**

This event is sent at the beginning of an app install or update and is used to track the very beginning of the install or update process.

The following fields are available:

- **ProductId** The product ID of the app that is being updated or installed.
- **PFN** The Package Family Name of the app that is being installed or updated.

#### **Microsoft.Windows.StoreAgent.Telemetry.InstallOperationRequest**

This event happens at the beginning of the install process when an app update or new app is installed. It's used to help keep Windows up-to-date and secure.



The following fields are available:

- **CatalogId** If this product is from a private catalog, the Store Product ID for the product being installed.
- **BundleId** The identity of the build associated with this product.
- **Skuld** Specific edition ID being installed.
- **ProductId** The Store Product ID for the product being installed.
- **VolumePath** The disk path of the installation.

#### **Microsoft.Windows.StoreAgent.Telemetry.PauseInstallation**

This event is sent when a product install or update is paused either by a user or the system. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **RelatedCV** Correlation Vector of a previous performed action on this product.
- **IsRemediation** Is this repairing a previous installation?
- **PreviousHResult** The result code of the last action performed before this operation.
- **ProductId** The Store Product ID for the product being installed.
- **IsUpdate** Is this an update?
- **PreviousInstallState** Previous state before the installation or update was paused.
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **AttemptNumber** The total number of retry attempts before it was canceled.
- **IsInteractive** Is this user requested?
- **BundleId** The identity of the build associated with this product.
- **PFN** The Product Full Name.
- **IsMandatory** Is this a mandatory update?
- **SystemAttemptNumber** The total number of system attempts.
- **UserAttemptNumber** The total number of user attempts.
- **IsRestore** Is this restoring previously acquired content?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **IsBundle** Is this a bundle?
- **WUContentId** The Windows Update content ID
- **AggregatedPackageFullNames** The names of all packages to be downloaded and installed.

#### **Microsoft.Windows.StoreAgent.Telemetry.ResumeInstallation**

This event happens when a product install or update is resumed either by a user or the system. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **RelatedCV** Correlation Vector for the original install before it was resumed.
- **AttemptNumber** The number of retry attempts before it was canceled.
- **BundleId** The identity of the build associated with this product.
- **PreviousHResult** The previous HResult error code.
- **ClientAppId** The identity of the app that initiated this operation.
- **CategoryId** The identity of the package or packages being installed.
- **PFN** The name of the package or packages requested for install.
- **IsUpdate** Is this an update?
- **PreviousInstallState** Previous state before the installation was paused.
- **IsRemediation** Is this repairing a previous installation?

- **IsInteractive** Is this user requested?
- **ProductId** The Store Product ID for the product being installed.
- **IsMandatory** Is this a mandatory update?
- **SystemAttemptNumber** The total number of system attempts.
- **UserAttemptNumber** The total number of user attempts.
- **IsRestore** Is this restoring previously acquired content?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **IsBundle** Is this a bundle?
- **WUContentId** The Windows Update content ID
- **AggregatedPackageFullNames** The names of all packages to be downloaded and installed.
- **IsUserRetry** Did the user initiate the retry?
- **HResult** The result code of the last action performed before this operation.

#### **Microsoft.Windows.StoreAgent.Telemetry.ResumeOperationRequest**

This event happens when a product install or update is resumed by a user and on installation retries. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **ProductId** The Store Product ID for the product being installed.

#### **Microsoft.Windows.StoreAgent.Telemetry.SearchForUpdateOperationRequest**

This event is sent when searching for update packages to install. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **ProductId** The Store Product ID for the product being installed.
- **Skuld** Specific edition of the app being updated.
- **CatalogId** The Store Product ID for the product being installed.

#### **Microsoft.Windows.StoreAgent.Telemetry.UpdateAppOperationRequest**

This event happens an app for a user needs to be updated. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **PFamN** The name of the product that is requested for update.

## Windows Update Delivery Optimization events

#### **Microsoft.OSG.DU.DeliveryOptClient.DownloadCanceled**

This event describes when a download was canceled with Delivery Optimization. It's used to understand and address problems regarding downloads.

The following fields are available:

- **bytesFromIntPeers** The number of bytes received from peers not in the same LAN or in the same group.
- **fileID** The ID of the file being downloaded.
- **sessionID** The ID of the file download session.
- **scenarioID** The ID of the scenario.
- **bytesFromCDN** The number of bytes received from a CDN source.
- **updateID** The ID of the update being downloaded.
- **background** Is the download being done in the background?
- **bytesFromPeers** The number of bytes received from a peer in the same LAN.

- **clientTelld** A random number used for device sampling.
- **bytesFromGroupPeers** The number of bytes received from a peer in the same group.
- **errorCode** The error code that was returned.
- **doErrorCode** The Delivery Optimization error code that was returned.
- **cdnErrorCodes** A list of CDN connection errors since the last FailureCDNCommunication event.
- **cdnErrorCounts** The number of times each error in cdnErrorCodes was encountered.
- **experimentId** When running a test, this is used to correlate events that are part of the same test.
- **isVpn** Is the device connected to a Virtual Private Network?
- **usedMemoryStream** Did the download use memory streaming?

### Microsoft.OSG.DU.DeliveryOptClient.DownloadCompleted

This event describes when a download has completed with Delivery Optimization. It's used to understand and address problems regarding downloads.

The following fields are available:

- **sessionID** The ID of the download session.
- **scenarioID** The ID of the scenario.
- **bytesFromIntPeers** The number of bytes received from peers not in the same LAN or in the same domain group.
- **updateID** The ID of the update being downloaded.
- **fileSize** The size of the file being downloaded.
- **bytesFromCDN** The number of bytes received from a CDN source.
- **fileID** The ID of the file being downloaded.
- **background** Is the download a background download?
- **bytesFromPeers** The number of bytes received from a peer in the same LAN.
- **totalTime** How long did the download take (in seconds)?
- **restrictedUpload** Is the upload restricted?
- **clientTelld** A random number used for device sampling.
- **bytesFromGroupPeers** The number of bytes received from a peer in the same domain group.
- **downloadMode** The download mode used for this file download session.
- **doErrorCode** The Delivery Optimization error code that was returned.
- **numPeers** The total number of peers used for this download.
- **cdnConnectionCount** The total number of connections made to the CDN.
- **lanConnectionCount** The total number of connections made to peers in the same LAN.
- **groupConnectionCount** The total number of connections made to peers in the same group.
- **internetConnectionCount** The total number of connections made to peers not in the same LAN or the same group.
- **cdnIp** The IP address of the source CDN.
- **downloadBps** The maximum measured available download bandwidth (in bytes per second).
- **uploadBps** The maximum measured available upload bandwidth (in bytes per second).
- **downloadUsageBps** The download speed (in bytes per second).
- **uploadUsageBps** The upload speed (in bytes per second).
- **totalTimeMs** Duration of the download (in seconds).
- **cdnErrorCodes** A list of CDN connection errors since the last FailureCDNCommunication event.
- **cdnErrorCounts** The number of times each error in cdnErrorCodes was encountered.
- **bytesRequested** The total number of bytes requested for download.
- **experimentId** When running a test, this is used to correlate with other events that are part of the same test.

- **isVpn** Is the device connected to a Virtual Private Network?
- **usedMemoryStream** Did the download use memory streaming?

### Microsoft.OSG.DU.DeliveryOptClient.DownloadPaused

This event represents a temporary suspension of a download with Delivery Optimization. It's used to understand and address problems regarding downloads.

The following fields are available:

- **updateID** The ID of the update being paused.
- **errorCode** The error code that was returned.
- **scenarioID** The ID of the scenario.
- **background** Is the download a background download?
- **sessionID** The ID of the download session.
- **clientTelId** A random number used for device sampling.
- **reasonCode** The reason for pausing the download.
- **fileID** The ID of the file being paused.
- **experimentId** When running a test, this is used to correlate with other events that are part of the same test.
- **isVpn** Is the device connected to a Virtual Private Network?

### Microsoft.OSG.DU.DeliveryOptClient.DownloadStarted

This event describes the start of a new download with Delivery Optimization. It's used to understand and address problems regarding downloads.

The following fields are available:

- **experimentId** When running a test, this is used to correlate with other events that are part of the same test.
- **errorCode** The error code that was returned.
- **doErrorCode** The Delivery Optimization error code that was returned.
- **peerID** The ID for this Delivery Optimization client.
- **doClientVersion** The version of the Delivery Optimization client.
- **jobID** The ID of the Windows Update job.
- **sessionID** The ID of the download session.
- **updateID** The ID of the update being downloaded.
- **scenarioID** The ID of the scenario.
- **fileID** The ID of the file being downloaded.
- **cdnUrl** The URL of the CDN.
- **filePath** The path where the file will be written.
- **groupID** ID for the group.
- **background** Is the download a background download?
- **downloadMode** The download mode used for this file download session.
- **minFileSizePolicy** The minimum content file size policy to allow the download using Peering.
- **diceRoll** The dice roll value used in sampling events.
- **deviceProfile** Identifies the usage or form factor. Example: Desktop or Xbox
- **isVpn** Is the device connected to a Virtual Private Network?
- **usedMemoryStream** Did the download use memory streaming?
- **minDiskSizePolicyEnforced** Is the minimum disk size enforced via policy?
- **minDiskSizeGB** The minimum disk size (in GB) required for Peering.
- **clientTelId** A random number used for device sampling.
- **costFlags** A set of flags representing network cost.

### Microsoft.OSG.DU.DeliveryOptClient.FailureCdnCommunication

This event represents a failure to download from a CDN with Delivery Optimization. It's used to understand and address problems regarding downloads.

The following fields are available:

- **experimentId** When running a test, this is used to correlate with other events that are part of the same test.
- **fileId** The ID of the file being downloaded.
- **errorCode** The error code that was returned.
- **httpStatusCode** The HTTP status code returned by the CDN.
- **errorCount** The total number of times this error code was seen since the last FailureCdnCommunication event was encountered.
- **sessionId** The ID of the download session.
- **cdnUrl** The URL of the CDN.
- **cdnIp** The IP address of the CDN.
- **cdnHeaders** The HTTP headers returned by the CDN.
- **clientTelId** A random number used for device sampling.
- **isHeadRequest** The type of HTTP request that was sent to the CDN. Example: HEAD or GET
- **requestSize** The size of the range requested from the CDN.
- **responseSize** The size of the range response received from the CDN.

### Microsoft.OSG.DU.DeliveryOptClient.JobError

This event represents a Windows Update job error. It allows for investigation of top errors.

The following fields are available:

- **jobId** The Windows Update job ID.
- **fileId** The ID of the file being downloaded.
- **errorCode** The error code returned.
- **clientTelId** A random number used for device sampling.
- **experimentId** When running a test, this is used to correlate with other events that are part of the same test.

## Windows Update events

### Microsoft.Windows.Update.DataMigrationFramework.DmfMigrationCompleted

This event sends data collected at the end of the Data Migration Framework (DMF) and parameters involved in its invocation, to help keep Windows up to date.

The following fields are available:

- **MigrationEndtime** A system timestamp of when the DMF migration completed.
- **UpdateIds** A collection of GUIDs for updates that are associated with the DMF session.
- **WuClientid** The GUID of the Windows Update client responsible for triggering the DMF migration.
- **MigrationDurationinmilliseconds** How long the DMF migration took (in milliseconds).
- **RevisionNumbers** A collection of revision numbers for the updates associated with the DMF session.

### Microsoft.Windows.Update.DataMigrationFramework.DmfMigrationStarted

This event sends data collected at the beginning of the Data Migration Framework (DMF) and parameters involved in its invocation, to help keep Windows up to date.

The following fields are available:

- **UpdateIds** A collection of GUIDs identifying the upgrades that are running.

- **MigrationStarttime** The timestamp representing the beginning of the DMF migration.
- **MigrationOEMphases** The number of OEM-authored migrators scheduled to be ran by DMF for this upgrade.
- **WuClientid** The GUID of the Windows Update client invoking DMF.
- **MigrationMicrosoftphases** The number of Microsoft-authored migrators scheduled to be ran by DMF for this upgrade.
- **RevisionNumbers** A collection of the revision numbers associated with the UpdateIds.

#### **Microsoft.Windows.Update.DataMigrationFramework.MigratorResult**

This event sends DMF migrator data to help keep Windows up to date.

The following fields are available:

- **MigratorGuid** A GUID identifying the migrator that just completed.
- **RunDurationInSeconds** The time it took for the migrator to complete.
- **CurrentStep** This is the last step the migrator reported before returning a result. This tells us how far through the individual migrator the device was before failure.
- **MigratorName** The name of the migrator that just completed.
- **MigratorId** A GUID identifying the migrator that just completed.
- **ErrorCode** The result (as an HRESULT) of the migrator that just completed.
- **TotalSteps** Migrators report progress in number of completed steps against the total steps. This is the total number of steps.

#### **Microsoft.Windows.Update.Orchestrator.CommitFailed**

This events tracks when a device needs to restart after an update but did not.

The following fields are available:

- **wuDeviceid** The Windows Update device GUID.
- **errorCode** The error code that was returned.

#### **Microsoft.Windows.Update.Orchestrator.Detection**

This event sends launch data for a Windows Update scan to help keep Windows up to date.

The following fields are available:

- **wuDeviceid** Unique device ID used by Windows Update.
- **revisionNumber** Update revision number.
- **eventScenario** End to end update session ID, or indicates the purpose of sending this event - whether because the software distribution just started installing content, or whether it was cancelled, succeeded, or failed.
- **deferReason** Reason why the device could not check for updates.
- **detectionBlockreason** Reason for detection not completing.
- **interactive** Identifies if session is User Initiated.
- **updateId** Update ID.
- **detectionDeferreason** A log of deferral reasons for every update state.
- **flightID** A unique update ID.
- **updateScenarioType** The update session type.
- **errorCode** The returned error code.

#### **Microsoft.Windows.Update.Orchestrator.Download**

This event sends launch data for a Windows Update download to help keep Windows up to date.

The following fields are available:

- **detectionDeferreason** Reason for download not completing
- **wuDeviceid** Unique device ID used by Windows Update.
- **interactive** Identifies if session is user initiated.
- **revisionNumber** Update revision number.
- **deferReason** Reason for download not completing
- **updateId** Update ID.
- **eventScenario** End to end update session ID.
- **errorCode** An error code represented as a hexadecimal value
- **flightID** Unique update ID.
- **updateScenarioType** The update session type.

#### **Microsoft.Windows.Update.Orchestrator.FlightInapplicable**

This event sends data on whether the update was applicable to the device, to help keep Windows up to date.

The following fields are available:

- **updateId** Unique Update ID
- **revisionNumber** Revision Number of the Update
- **UpdateStatus** Integer that describes Update state
- **EventPublishedTime** time that the event was generated
- **wuDeviceid** Unique Device ID
- **flightID** Unique Update ID
- **updateScenarioType** The update session type.

#### **Microsoft.Windows.Update.Orchestrator.InitiatingReboot**

This event sends data about an Orchestrator requesting a reboot from power management to help keep Windows up to date.

The following fields are available:

- **revisionNumber** Revision number of the update.
- **EventPublishedTime** Time of the event.
- **updateId** Update ID.
- **wuDeviceid** Unique device ID used by Windows Update.
- **flightID** Unique update ID
- **interactive** Indicates the reboot initiation stage of the update process was entered as a result of user action or not.
- **rebootOutsideOfActiveHours** Indicates the timing that the reboot was to occur to ensure the correct update process and experience is provided to keep Windows up to date.
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date.
- **updateScenarioType** The update session type.

#### **Microsoft.Windows.Update.Orchestrator.Install**

This event sends launch data for a Windows Update install to help keep Windows up to date.

The following fields are available:

- **eventScenario** End to end update session ID.
- **deferReason** Reason for install not completing.
- **interactive** Identifies if session is user initiated.
- **wuDeviceid** Unique device ID used by Windows Update.

- **batteryLevel** Current battery capacity in mWh or percentage left.
- **installCommitFailedTime** The time it took for a reboot to happen but the upgrade failed to progress.
- **errorCode** The error code represented by a hexadecimal value.
- **updateId** Update ID.
- **revisionNumber** Update revision number.
- **flightID** Unique update ID
- **installRebootInitiateTime** The time it took for a reboot to be attempted.
- **flightUpdate** Flight update
- **minutesToCommit** The time it took to install updates.
- **ForcedRebootReminderSet** A boolean value that indicates if a forced reboot will happen for updates.
- **rebootOutsideOfActiveHours** Indicates the timing that the reboot was to occur to ensure the correct update process and experience is provided to keep Windows up to date.
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date.
- **updateScenarioType** The update session type.

#### **Microsoft.Windows.Update.Orchestrator.PostInstall**

This event sends data about lite stack devices (mobile, IOT, anything non-PC) immediately before data migration is launched to help keep Windows up to date.

The following fields are available:

- **wuDeviceid** Unique device ID used by Windows Update.
- **eventScenario** End to end update session ID.
- **sessionType** Interactive vs. Background.
- **bundleRevisionnumber** Bundle revision number.
- **batteryLevel** Current battery capacity in mWh or percentage left.
- **bundleId** Update grouping ID.
- **errorCode** Hex code for the error message, to allow lookup of the specific error.
- **flightID** Unique update ID.

#### **Microsoft.Windows.Update.Orchestrator.RebootFailed**

This event sends information about whether an update required a reboot and reasons for failure to help keep Windows up to date.

The following fields are available:

- **updateId** Update ID.
- **batteryLevel** Current battery capacity in mWh or percentage left.
- **RebootResults** Hex code indicating failure reason. Typically, we expect this to be a specific USO generated hex code.
- **installRebootDeferreason** Reason for reboot not occurring.
- **revisionNumber** Update revision number.
- **EventPublishedTime** The time that the reboot failure occurred.
- **deferReason** Reason for install not completing.
- **wuDeviceid** Unique device ID used by Windows Update.
- **flightID** Unique update ID.
- **rebootOutsideOfActiveHours** Indicates the timing that the reboot was to occur to ensure the correct update process and experience is provided to keep Windows up to date.
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date.



- **updateScenarioType** The update session type.

#### **Microsoft.Windows.Update.Orchestrator.RestoreRebootTask**

This event sends data indicating that a reboot task is missing unexpectedly on a device and the task is restored because a reboot is still required, to help keep Windows up to date.

The following fields are available:

- **RebootTaskRestoredTime** Time at which this reboot task was restored.
- **wuDeviceid** Device id on which the reboot is restored
- **revisionNumber** Update revision number.
- **updateId** Update ID.

#### **Microsoft.Windows.Update.Orchestrator.SystemNeeded**

This event sends data about why a device is unable to reboot, to help keep Windows up to date.

The following fields are available:

- **eventScenario** End to end update session ID.
- **wuDeviceid** Unique device ID used by Windows Update.
- **systemNeededReason** Reason ID
- **updateId** Update ID.
- **revisionNumber** Update revision number.
- **rebootOutsideOfActiveHours** Indicates the timing that the reboot was to occur to ensure the correct update process and experience is provided to keep Windows up to date.
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date.
- **updateScenarioType** The update session type.

#### **Microsoft.Windows.Update.Orchestrator.UpdatePolicyCacheRefresh**

This event sends data on whether Update Management Policies were enabled on a device, to help keep Windows up to date.

The following fields are available:

- **wuDeviceid** Unique device ID used by Windows Update.
- **policyCacherefreshtime** Refresh time
- **policiesNamevaluesource** Policy Name
- **updateInstalluxsetting** This shows whether a user has set policies via UX option
- **configuredPoliciescount** Policy Count

#### **Microsoft.Windows.Update.Orchestrator.UpdateRebootRequired**

This event sends data about whether an update required a reboot to help keep Windows up to date.

The following fields are available:

- **updateId** Update ID.
- **revisionNumber** Update revision number.
- **wuDeviceid** Unique device ID used by Windows Update.
- **flightID** Unique update ID.
- **interactive** Indicates the reboot initiation stage of the update process was entered as a result of user action or not.
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date.

- **updateScenarioType** The update session type.

### **Microsoft.Windows.Update.UpdateStackServicing.CheckForUpdates**

This event sends data about the UpdateStackServicing check for updates, to help keep Windows up to date.

The following fields are available:

- **EventScenario** The scenario of the event. Example: Started, Failed, or Succeeded
- **StatusCode** The HRESULT code of the operation.
- **CallerApplicationName** The name of the USS scheduled task. Example UssScheduled or UssBoot
- **ClientVersion** The version of the client.
- **EventInstanceID** The USS session ID.
- **WUDeviceID** The Windows Update device ID.
- **ServiceGuid** The GUID of the service.
- **BspVersion** The version of the BSP.
- **OemName** The name of the manufacturer.
- **DeviceName** The name of the device.
- **CommercializationOperator** The name of the operator.
- **DetectionVersion** The string returned from the GetDetectionVersion export of the downloaded detection DLL.

### **Microsoft.Windows.Update.Ux.MusNotification.RebootNoLongerNeeded**

This event is sent when a security update has successfully completed.

The following fields are available:

- **UtcTime** The Coordinated Universal Time that the restart was no longer needed.

### **Microsoft.Windows.Update.Ux.MusNotification.RebootScheduled**

This event sends data about a required reboot that is scheduled with no user interaction, to help keep Windows up to date.

The following fields are available:

- **updateId** Update ID of the update that is getting installed with this reboot.
- **ScheduledRebootTime** Time of the scheduled reboot.
- **wuDeviceid** Unique device ID used by Windows Update.
- **revisionNumber** Revision number of the update that is getting installed with this reboot.
- **forcedreboot** True, if a reboot is forced on the device. False, otherwise.
- **rebootArgument** Argument for the reboot task. It also represents specific reboot related action.
- **rebootScheduledByUser** True, if a reboot is scheduled by user. False, if a reboot is scheduled automatically.
- **activeHoursApplicable** True, If Active Hours applicable on this device. False, otherwise.
- **rebootOutsideOfActiveHours** True, if a reboot is scheduled outside of active hours. False, otherwise.
- **rebootState** The state of the reboot.

### **Microsoft.Windows.Update.Ux.MusNotification.ToastDisplayedToScheduleReboot**

This event is sent when a toast notification is shown to the user about scheduling a device restart.

The following fields are available:

- **UtcTime** The Coordinated Universal Time when the toast notification was shown.

### **Microsoft.Windows.Update.Ux.MusUpdateSettings.RebootScheduled**

This event sends basic information for scheduling a device restart to install security updates. It's used to help keep

Windows up-to-date.

The following fields are available:

- **ScheduledRebootTime** The time that the device was restarted.
- **updateId** The Windows Update device GUID.
- **revisionNumber** The revision number of the OS being updated.
- **wuDeviceid** The Windows Update device GUID.
- **forcedreboot** Is the restart that's being scheduled a forced restart?
- **rebootArgument** The arguments that are passed to the OS for the restarted.
- **rebootScheduledByUser** Was the restart scheduled by the user? If the value is false, the restart was scheduled by the device.
- **activeHoursApplicable** Is the restart respecting Active Hours?
- **rebootOutsideOfActiveHours** Was the restart scheduled outside of Active Hours?
- **rebootState** The state of the restart.

## Winlogon events

### **Microsoft.Windows.Security.Winlogon.SetupCompleteLogon**

This event signals the completion of the setup process. It happens only once during the first logon.

# Windows 10 enhanced diagnostic data events and fields used by Windows Analytics

5/30/2018 • 9 minutes to read • [Edit Online](#)

## Applies to

- Windows 10, version 1709 and newer

Windows Analytics Device Health reports are powered by diagnostic data not included in the Basic level. This includes crash reports and certain OS diagnostic data events. Organizations sending Enhanced or Full level diagnostic data were able to participate in Device Health, but some organizations which required detailed event and field level documentation were unable to move from Basic to Enhanced.

In Windows 10, version 1709, we introduce a new feature: "Limit Enhanced diagnostic data to the minimum required by Windows Analytics". When enabled, this feature limits the operating system diagnostic data events included in the Enhanced level to only those described below. Note that the Enhanced level also includes limited crash reports, which are not described below. For more information on the Enhanced level, see [Configure Windows diagnostic data in your organization](#).

## KernelProcess.AppStateChangeSummary

This event summarizes application usage and performance characteristics to help Microsoft improve performance and reliability. Organizations can use this event with Windows Analytics to gain insights into application reliability.

The following fields are available:

- **CommitChargeAtExit\_Sum:** Total memory commit charge for a process when it exits
- **CommitChargePeakAtExit\_Sum:** Total peak memory commit charge for a process when it exits
- **ContainerId:** Server Silo Container ID
- **CrashCount:** Number of crashes for a process instance
- **CycleCountAtExit\_Sum:** Total processor cycles for a process when it exited
- **ExtraInfoFlags:** Flags indicating internal states of the logging
- **GhostCount\_Sum:** Total number of instances where the application stopped responding
- **HandleCountAtExit\_Sum:** Total handle count for a process when it exits
- **HangCount\_Max:** Maximum number of hangs detected
- **HangCount\_Sum:** Total number of application hangs detected
- **HardFaultCountAtExit\_Sum:** Total number of hard page faults detected for a process when it exits
- **HeartbeatCount:** Heartbeats logged for this summary
- **HeartbeatSuspendedCount:** Heartbeats logged for this summary where the process was suspended
- **LaunchCount:** Number of process instances started
- **LicenseType:** Reserved for future use
- **ProcessDurationMS\_Sum:** Total duration of wall clock process instances
- **ReadCountAtExit\_Sum:** Total IO reads for a process when it exited
- **ReadSizeInKBAtExit\_Sum:** Total IO read size for a process when it exited
- **ResumeCount:** Number of times a process instance has resumed
- **RunningDurationMS\_Sum:** Total uptime
- **SuspendCount:** Number of times a process instance was suspended

- **TargetAppId:** Application identifier
- **TargetAppType:** Application type
- **TargetAppVer:** Application version
- **TerminateCount:** Number of times a process terminated
- **WriteCountAtExit\_Sum:** Total number of IO writes for a process when it exited
- **WriteSizeInKBAtExit\_Sum:** Total size of IO writes for a process when it exited

## Microsoft.OSG.OSS.CredProvFramework.ReportResultStop

This event indicates the result of an attempt to authenticate a user with a credential provider. It helps Microsoft to improve logon reliability. Using this event with Windows Analytics can help organizations monitor and improve logon success for different methods (for example, biometric) on managed devices.

The following fields are available:

- **CredTileProviderId:** ID of the Credential Provider
- **IsConnectedUser:** Flag indicating whether a user is connected or not
- **IsPLAPTile:** Flag indicating whether this credential tile is a pre-logon access provider or not
- **IsRemoteSession:** Flag indicating whether the session is remote or not
- **IsV2CredProv:** Flag indicating whether the credential provider of V2 or not
- **OptionalStatusText:** Status text
- **ProcessImage:** Image path to the process
- **ProviderId:** Credential provider ID
- **ProviderStatusIcon:** Indicates which status icon should be displayed
- **ReturnCode:** Output of the ReportResult function
- **SessionId:** Session identifier
- **Sign-in error status:** The sign-in error status
- **SubStatus:** Sign-in error sub-status
- **UserTag:** Count of the number of times a user has selected a provider

## Microsoft.Windows.Kernel.Power.OSStateChange

This event denotes the transition between operating system states (e.g., On, Off, Sleep, etc.). By using this event with Windows Analytics, organizations can use this to monitor reliability and performance of managed devices

The following fields are available:

- **AcPowerOnline:** If "TRUE," the device is using AC power. If "FALSE," the device is using battery power.
- **ActualTransitions:** The number of transitions between operating system states since the last system boot
- **BatteryCapacity:** Maximum battery capacity in mWh
- **BatteryCharge:** Current battery charge as a percentage of total capacity
- **BatteryDischarging:** Flag indicating whether the battery is discharging or charging
- **BootId:** Total boot count since the operating system was installed
- **BootTimeUTC:** Date and time of a particular boot event (identified by BootId)
- **EnergyChangeV2:** A snapshot value in mWh reflecting a change in power usage
- **EnergyChangeV2Flags:** Flags for disambiguating EnergyChangeV2 context
- **EventSequence:** A sequential number used to evaluate the completeness of the data
- **LastStateTransition:** ID of the last operating system state transition
- **LastStateTransitionSub:** ID of the last operating system sub-state transition
- **StateDurationMS:** Number of milliseconds spent in the last operating system state

- **StateTransition:** ID of the operating system state the system is transitioning to
- **StateTransitionSub:** ID of the operating system sub-state the system is transitioning to
- **TotalDurationMS:** Total time (in milliseconds) spent in all states since the last boot
- **TotalUptimeMS:** Total time (in milliseconds) the device was in Up or Running states since the last boot
- **TransitionsToOn:** Number of transitions to the Powered On state since the last boot
- **UptimeDeltaMS:** Total time (in milliseconds) added to Uptime since the last event

## Microsoft.Windows.LogonController.LogonAndUnlockSubmit

Sends details of the user attempting to sign into or unlock the device.

The following fields are available:

- **isSystemManagedAccount:** Indicates if the user's account is System Managed
- **isUnlockScenario:** Flag indicating whether the event is a Logon or an Unlock
- **PartA\_UserSid:** The security identifier of the user
- **userType:** Indicates the user type: 0 = unknown; 1 = local; 2 = Active Directory domain user; 3 = Microsoft Account; 4 = Azure Active Directory user

## Microsoft.Windows.LogonController.SignInFailure

Sends details about any error codes detected during a failed sign-in.

The following fields are available:

- **ntsStatus:** The NTSTATUS error code status returned from an attempted sign-in
- **ntsSubstatus:** The NTSTATUS error code sub-status returned from an attempted sign-in

## Microsoft.Windows.Security.Biometrics.Service.BioServiceActivityCapture

Indicates that a biometric capture was compared to known templates

The following fields are available:

- **captureDetail:** Result of biometric capture, either matched to an enrollment or an error
- **captureSuccessful:** Indicates whether a biometric capture was successfully matched or not
- **hardwareId:** ID of the sensor that collected the biometric capture
- **isSecureSensor:** Flag indicating whether a biometric sensor was in enhanced security mode
- **isTrustletRunning:** Indicates whether an enhanced security component is currently running
- **isVsmCfg:** Flag indicating whether virtual secure mode is configured or not

## Microsoft.Windows.Security.Certificates.PinRulesCaCertUsedAnalytics

The Microsoft.Windows.Security.Certificates.Pin\*Analytics events summarize which server certificates the client encounters. By using this event with Windows Analytics, organizations can use this to determine potential scope and impact of pending certificate revocations or expirations.

The following fields are available:

- **certBinary:** Binary blob of public certificate as presented to the client (does not include any private keys)
- **certThumbprint:** Certificate thumbprint

## Microsoft.Windows.Security.Certificates.PinRulesCheckedAnalytics

The Microsoft.Windows.Security.Certificates.Pin\*Analytics events summarize which server certificates the client encounters. By using this event with Windows Analytics, organizations can use this to determine potential scope and impact of pending certificate revocations or expirations.

The following fields are available:

- **caThumbprints:** Intermediate certificate thumbprints
- **rootThumbprint:** Root certificate thumbprint
- **serverName:** Server name associated with the certificate
- **serverThumbprint:** Server certificate thumbprint
- **statusBits:** Certificate status

## Microsoft.Windows.Security.Certificates.PinRulesServerCertUsedAnalytics

The Microsoft.Windows.Security.Certificates.Pin\*Analytics events summarize which server certificates the client encounters. By using this event with Windows Analytics, organizations can use this to determine potential scope and impact of pending certificate revocations or expirations.

The following fields are available:

- **certBinary:** Binary blob of public certificate as presented to the client (does not include any private keys)
- **certThumbprint:** Certificate thumbprint

## Microsoft.Windows.Security.Winlogon.SystemBootStop

System boot has completed.

The following field is available:

- **ticksSinceBoot:** Duration of boot event (milliseconds)

## Microsoft.Windows.Shell.Desktop.LogonFramework.AllLogonTasks

This event summarizes the logon procedure to help Microsoft improve performance and reliability. By using this event with Windows Analytics organizations can help identify logon problems on managed devices.

The following fields are available:

- **isAadUser:** Indicates whether the current logon is for an Azure Active Directory account
- **isDomainUser:** Indicates whether the current logon is for a domain account
- **isMSA:** Indicates whether the current logon is for a Microsoft Account
- **logonOptimizationFlags:** Flags indicating optimization settings for this logon session
- **logonTypeFlags:** Flags indicating logon type (first logon vs. a later logon)
- **systemManufacturer:** Device manufacturer
- **systemProductName:** Device product name
- **wilActivity:** Indicates errors in the task to help Microsoft improve reliability.

## Microsoft.Windows.Shell.Desktop.LogonFramework.LogonTask

This event describes system tasks which are part of the user logon sequence and helps Microsoft to improve reliability.

The following fields are available:

- **isStartWaitTask:** Flag indicating whether the task starts a background task
- **isWaitMethod:** Flag indicating the task is waiting on a background task
- **logonTask:** Indicates which logon step is currently occurring
- **wilActivity:** Indicates errors in the task to help Microsoft improve reliability.

## Microsoft.Windows.Shell.Explorer.DesktopReady

Initialization of Explorer is complete.

## Microsoft-Windows-Security-EDP-Audit-ApplicationLearning.EdpAuditLogApplicationLearning

For a device subject to Windows Information Protection policy, learning events are generated when an app encounters a policy boundary (for example, trying to open a work document from a personal app). These events help the WIP administrator tune policy rules and prevent unnecessary user disruption.

The following fields are available:

- **actiontype:** Indicates what type of resource access the app was attempting (for example, opening a local document vs. a network resource) when it encountered a policy boundary. Useful for Windows Information Protection administrators to tune policy rules.
- **appIdType:** Based on the type of application, this indicates what type of app rule a Windows Information Protection administrator would need to create for this app.
- **appName:** App that triggered the event
- **status:** Indicates whether errors occurred during WIP learning events

## Win32kTraceLogging.AppInteractivitySummary

Summarizes which app windows are being used (for example, have focus) to help Microsoft improve compatibility and user experience. Also helps organizations (by using Windows Analytics) to understand and improve application reliability on managed devices.

The following fields are available:

- **AggregationDurationMS:** Actual duration of aggregation period (in milliseconds)
- **AggregationFlags:** Flags denoting aggregation settings
- **AggregationPeriodMS:** Intended duration of aggregation period (in milliseconds)
- **AggregationStartTime:** Start date and time of AppInteractivity aggregation
- **AppId:** Application ID for usage
- **AppSessionId:** GUID identifying the application's usage session
- **AppVersion:** Version of the application that produced this event
- **AudioInMS:** Audio capture duration (in milliseconds)
- **AudioOutMS:** Audio playback duration (in milliseconds)
- **BackgroundMouseSec:** Indicates that there was a mouse hover event while the app was in the background
- **BitPeriodMS:** Length of the period represented by InFocusBitmap
- **CommandLineHash:** A hash of the command line
- **CompositionDirtyGeneratedSec:** Represents the amount of time (in seconds) during which the active app reported that it had an update
- **CompositionDirtyPropagatedSec:** Total time (in seconds) that a separate process with visuals hosted in an app signaled updates
- **CompositionRenderedSec:** Time (in seconds) that an app's contents were rendered



- **EventSequence:** [need more info]
- **FocusLostCount:** Number of times that an app lost focus during the aggregation period
- **GameInputSec:** Time (in seconds) there was user input using a game controller
- **HidInputSec:** Time (in seconds) there was user input using devices other than a game controller
- **InFocusBitmap:** Series of bits representing application having and losing focus
- **InFocusDurationMS:** Total time (in milliseconds) the application had focus
- **InputSec:** Total number of seconds during which there was any user input
- **InteractiveTimeoutPeriodMS:** Total time (in milliseconds) that inactivity expired interactivity sessions
- **KeyboardInputSec:** Total number of seconds during which there was keyboard input
- **MonitorFlags:** Flags indicating app use of individual monitor(s)
- **MonitorHeight:** Number of vertical pixels in the application host monitor resolution
- **MonitorWidth:** Number of horizontal pixels in the application host monitor resolution
- **MouseInputSec:** Total number of seconds during which there was mouse input
- **NewProcessCount:** Number of new processes contributing to the aggregate
- **PartATransform\_AppSessionGuidToUserSid:** Flag which influences how other parts of the event are constructed
- **PenInputSec:** Total number of seconds during which there was pen input
- **SpeechRecognitionSec:** Total number of seconds of speech recognition
- **SummaryRound:** Incrementing number indicating the round (batch) being summarized
- **TargetAsId:** Flag which influences how other parts of the event are constructed
- **TotalUserOrDisplayActiveDurationMS:** Total time the user or the display was active (in milliseconds)
- **TouchInputSec:** Total number of seconds during which there was touch input
- **UserActiveDurationMS:** Total time that the user was active including all input methods
- **UserActiveTransitionCount:** Number of transitions in and out of user activity
- **UserOrDisplayActiveDurationMS:** Total time the user was using the display
- **ViewFlags:** Flags denoting properties of an app view (for example, special VR view or not)
- **WindowFlags:** Flags denoting runtime properties of an app window
- **WindowHeight:** Number of vertical pixels in the application window
- **WindowWidth:** Number of horizontal pixels in the application window

# Windows 10, version 1709 and newer diagnostic data for the Full level

5/22/2018 • 25 minutes to read • [Edit Online](#)

Applies to:

- Windows 10, version 1803
- Windows 10, version 1709

Microsoft uses Windows diagnostic data to keep Windows secure and up-to-date, troubleshoot problems, and make product improvements. For users who have turned on "Tailored experiences", it can also be used to offer you personalized tips, ads, and recommendations to enhance Microsoft products and services for your needs. This article describes all types of diagnostic data collected by Windows at the Full level (inclusive of data collected at Basic), with comprehensive examples of data we collect per each type. For additional, detailed technical descriptions of Basic data items, see [Windows 10, version 1803 Basic level diagnostic events and fields](#).

In addition, this article provides references to equivalent definitions for the data types and examples from [ISO/IEC 19944:2017 Information technology -- Cloud computing -- Cloud services and devices: Data flow, data categories and data use](#). Each data type also has a Data Use statement, for diagnostics and for Tailored experiences on the device, using the terms as defined by the standard. These Data Use statements define the purposes for which Microsoft processes each type of Windows diagnostic data, using a uniform set of definitions referenced at the end of this document and based on the ISO standard. Reference to the ISO standard provides additional clarity about the information collected, and allows easy comparison with other services or guidance that also references the standard.

The data covered in this article is grouped into the following types:

- Common data (diagnostic header information)
- Device, Connectivity, and Configuration data
- Product and Service Usage data
- Product and Service Performance data
- Software Setup and Inventory data
- Browsing History data
- Inking, Typing, and Speech Utterance data

## Common data

Most diagnostic events contain a header of common data. In each example, the info in parentheses provides the equivalent definition for ISO/IEC 19944:2017.

**Data Use for Common data** Header data supports the use of data associated with all diagnostic events. Therefore, Common data is used to [provide](#) Windows 10, and may be used to [improve](#), [personalize](#), [recommend](#), [offer](#), or [promote](#) Microsoft and third-party products and services, depending on the uses described in the **Data Use** statements for each data category.

### **Data Description for Common data type**

SUB-TYPE	DESCRIPTION AND EXAMPLES
Common Data	<p>Information that is added to most diagnostic events, if relevant and available:</p> <ul style="list-style-type: none"> <li>• Diagnostic level -- Basic or Full, Sample level -- for sampled data, what sample level is this device opted into (8.2.3.2.4 Observed Usage of the Service Capability)</li> <li>• Operating system name, version, build, and locale (8.2.3.2.2 Telemetry data)</li> <li>• Event collection time (8.2.3.2.2 Telemetry data)</li> <li>• User ID -- a unique identifier associated with the user's Microsoft Account (if one is used) or local account. The user's Microsoft Account identifier is not collected from devices configured to send Basic diagnostic data (8.2.5 Account data)</li> <li>• Xbox UserID (8.2.5 Account data)</li> <li>• Device ID -- This is not the user provided device name, but an ID that is unique for that device. (8.2.3.2.3 Connectivity data)</li> <li>• Device class -- Desktop, Server, or Mobile (8.2.3.2.3 Connectivity data)</li> <li>• Environment from which the event was logged -- Application ID of app or component that logged the event, Session GUID. Used to track events over a given period of time, such as the amount of time an app is running or between boots of the operating system (8.2.4 Cloud service provider data)</li> <li>• Diagnostic event name, Event ID, ETW opcode, version, schema signature, keywords, and flags (8.2.4 Cloud service provider data)</li> <li>• HTTP header information, including the IP address. This IP address is the source address that's provided by the network packet header and received by the diagnostics ingestion service (8.2.4 Cloud service provider data)</li> <li>• Various IDs that are used to correlate and sequence related events together (8.2.4 Cloud service provider data)</li> </ul>

## Device, Connectivity, and Configuration data

This type of data includes details about the device, its configuration and connectivity capabilities, and status. Device, Connectivity, and Configuration Data is equivalent to ISO/IEC 19944:2017, 8.2.3.2.3 Connectivity data.

### Data Use for Device, Connectivity, and Configuration data

#### For Diagnostics:

[Pseudonymized](#) Device, Connectivity, and Configuration data from Windows 10 is used by Microsoft to [provide](#) and [improve](#) Windows 10 and related Microsoft products and services. For example:

- Device, Connectivity, and Configuration data is used to understand the unique device characteristics that can contribute to an error experienced on the device, to identify patterns, and to more quickly resolve problems that impact devices with unique hardware, capabilities, or settings. For example:
  - Data about the use of cellular modems and their configuration on your devices is used to troubleshoot cellular modem issues.
  - Data about the use of USB hubs use and their configuration on your devices is used to troubleshoot USB hub issues.

- Data about the use of connected Bluetooth devices is used to troubleshoot compatibility issues with Bluetooth devices.
- Data about device properties, such as the operating system version and available memory, is used to determine whether the device is due to, and able to, receive a Windows update.
- Data about device peripherals is used to determine whether a device has installed drivers that might be negatively impacted by a Windows update.
- Data about which devices, peripherals, and settings are most-used by customers, is used to prioritize Windows 10 improvements to determine the greatest positive impact to the most Windows 10 users.

**With (optional) Tailored experiences:**

If a user has enabled Tailored experiences on the device, [Pseudonymized Device](#), Connectivity, and Configuration data from Windows 10 is used by Microsoft to [personalize](#), [recommend](#), and [offer](#) Microsoft products and services to Windows 10 users. Also, if a user has enabled Tailored experiences on the device, [Pseudonymized Device](#), Connectivity, and Configuration data from Windows 10 is used by Microsoft to [promote](#) third-party Windows apps, services, hardware, and peripherals to Windows 10 users. For example:

- Data about device properties and capabilities is used to provide tips about how to use or configure the device to get the best performance and user experience.
- Data about device capabilities, such as whether the device is pen-enabled, is used to recommend (Microsoft and third-party) apps that are appropriate for the device. These may be free or paid apps.

**Data Description for Device, Connectivity, and Configuration data type**

SUB-TYPE	DESCRIPTION AND EXAMPLES
Device properties	<p>Information about the operating system and device hardware, such as:</p> <ul style="list-style-type: none"> <li>● Operating system - version name, edition</li> <li>● Installation type, subscription status, and genuine operating system status</li> <li>● Processor architecture, speed, number of cores, manufacturer, and model</li> <li>● OEM details --manufacturer, model, and serial number</li> <li>● Device identifier and Xbox serial number</li> <li>● Firmware/BIOS operating system -- type, manufacturer, model, and version</li> <li>● Memory -- total memory, video memory, speed, and how much memory is available after the device has reserved memory</li> <li>● Storage -- total capacity and disk type</li> <li>● Battery -- charge capacity and InstantOn support</li> <li>● Hardware chassis type, color, and form factor</li> <li>● Is this a virtual machine?</li> </ul>

SUB-TYPE	DESCRIPTION AND EXAMPLES
Device capabilities	<p>Information about the specific device capabilities, such as:</p> <ul style="list-style-type: none"> <li>• Camera -- whether the device has a front facing camera, a rear facing camera, or both.</li> <li>• Touch screen -- Whether the device has a touch screen? If yes, how many hardware touch points are supported?</li> <li>• Processor capabilities -- CompareExchange128, LahfSahf, NX, PrefetchW, and SSE2</li> <li>• Trusted Platform Module (TPM) -- whether a TPM exists and if yes, what version</li> <li>• Virtualization hardware -- whether an IOMMU exists, whether it includes SLAT support, and whether virtualization is enabled in the firmware</li> <li>• Voice -- whether voice interaction is supported and the number of active microphones</li> <li>• Number of displays, resolutions, and DPI</li> <li>• Wireless capabilities</li> <li>• OEM or platform face detection</li> <li>• OEM or platform video stabilization and quality-level set</li> <li>• Advanced Camera Capture mode (HDR versus Low Light), OEM versus platform implementation, HDR probability, and Low Light probability</li> </ul>
Device preferences and settings	<p>Information about the device settings and user preferences, such as:</p> <ul style="list-style-type: none"> <li>• User Settings -- System, Device, Network &amp; Internet, Personalization, Cortana, Apps, Accounts, Time &amp; Language, Gaming, Ease of Access, Privacy, Update &amp; Security</li> <li>• User-provided device name</li> <li>• Whether device is domain-joined, or cloud-domain joined (for example, part of a company-managed network)</li> <li>• Hashed representation of the domain name</li> <li>• MDM (mobile device management) enrollment settings and status</li> <li>• BitLocker, Secure Boot, encryption settings, and status</li> <li>• Windows Update settings and status</li> <li>• Developer Unlock settings and status</li> <li>• Default app choices</li> <li>• Default browser choice</li> <li>• Default language settings for app, input, keyboard, speech, and display</li> <li>• App store update settings</li> <li>• Enterprise OrganizationID, Commercial ID</li> </ul>

SUB-TYPE	DESCRIPTION AND EXAMPLES
Device peripherals	<p>Information about the device peripherals, such as:</p> <ul style="list-style-type: none"> <li>• Peripheral name, device model, class, manufacturer, and description</li> <li>• Peripheral device state, install state, and checksum</li> <li>• Driver name, package name, version, and manufacturer</li> <li>• HWID - A hardware vendor-defined ID to match a device to a driver <a href="#">INF file</a></li> <li>• Driver state, problem code, and checksum</li> <li>• Whether driver is kernel mode, signed, and image size</li> </ul>
Device network info	<p>Information about the device network configuration, such as:</p> <ul style="list-style-type: none"> <li>• Network system capabilities</li> <li>• Local or Internet connectivity status</li> <li>• Proxy, gateway, DHCP, DNS details, and addresses</li> <li>• Whether it's a paid or free network</li> <li>• Whether the wireless driver is emulated</li> <li>• Whether it's access point mode-capable</li> <li>• Access point manufacturer, model, and MAC address</li> <li>• WDI Version</li> <li>• Name of networking driver service</li> <li>• Wi-Fi Direct details</li> <li>• Wi-Fi device hardware ID and manufacturer</li> <li>• Wi-Fi scan attempt and item counts</li> <li>• Whether MAC randomization is supported and enabled</li> <li>• Number of supported spatial streams and channel frequencies</li> <li>• Whether Manual or Auto-connect is enabled</li> <li>• Time and result of each connection attempt</li> <li>• Airplane mode status and attempts</li> <li>• Interface description provided by the manufacturer</li> <li>• Data transfer rates</li> <li>• Cipher algorithm</li> <li>• Mobile Equipment ID (IMEI) and Mobile Country Code (MCCO)</li> <li>• Mobile operator and service provider name</li> <li>• Available SSIDs and BSSIDs</li> <li>• IP Address type -- IPv4 or IPv6</li> <li>• Signal Quality percentage and changes</li> <li>• Hotspot presence detection and success rate</li> <li>• TCP connection performance</li> <li>• Miracast device names</li> <li>• Hashed IP address</li> </ul>

## Product and Service Usage data

This type of data includes details about the usage of the device, operating system, applications and services. Product and Service Usage data is equivalent to ISO/IEC 19944:2017, 8.2.3.2.4 Observed Usage of the Service Capability.

### Data Use for Product and Service Usage data

#### For Diagnostics:

[Pseudonymized](#) Product and Service Usage data from Windows 10 is used by Microsoft to [provide](#) and [improve](#) Windows 10 and related Microsoft product and services. For example:

- Data about the specific apps that are in-use when an error occurs is used to troubleshoot and repair issues with Windows features and Microsoft apps.
- Data about the specific apps that are most-used by customers, is used to prioritize Windows 10 improvements to determine the greatest positive impact to the most Windows 10 users.
- Data about whether devices have Suggestions turned off from the **Settings Phone** screen is to improve the Suggestions feature.
- Data about whether a user canceled the authentication process in their browser is used to help troubleshoot issues with and improve the authentication process.
- Data about when and what feature invoked Cortana is used to prioritize efforts for improvement and innovation in Cortana.
- Data about when a context menu in the photo app is closed is used to troubleshoot and improve the photo app.

**With (optional) Tailored experiences:**

If a user has enabled Tailored experiences on the device, [pseudonymized](#) Product and Service Usage data from Windows 10 is used by Microsoft to [personalize](#), [recommend](#), and [offer](#) Microsoft products and services to Windows 10 users. Also, if a user has enabled Tailored experiences on the device, [pseudonymized](#) Product and Service Usage data from Windows 10 is used by Microsoft to [promote](#) third-party Windows apps, services, hardware, and peripherals to Windows 10 users. For example:

- If data shows that a user has not used a particular feature of Windows, we may recommend that the user try that feature.
- Data about which apps are most-used on a device is used to provide recommendations for similar or complementary (Microsoft or third-party) apps. These may be free or paid apps.

**Data Description for Product and Service Usage data type**

SUB-TYPE	DESCRIPTION AND EXAMPLES
----------	--------------------------

SUB-TYPE	DESCRIPTION AND EXAMPLES
App usage	<p>Information about Windows and application usage, such as:</p> <ul style="list-style-type: none"> <li>• Operating system component and app feature usage</li> <li>• User navigation and interaction with app and Windows features. This could potentially include user input, such as name of a new alarm set, user menu choices, or user favorites</li> <li>• Time of and count of app and component launches, duration of use, session GUID, and process ID</li> <li>• App time in various states -- running in the foreground or background, sleeping, or receiving active user interaction</li> <li>• User interaction method and duration -- whether the user used a keyboard, mouse, pen, touch, speech, or game controller, and for how long</li> <li>• Cortana launch entry point and reason</li> <li>• Notification delivery requests and status</li> <li>• Apps used to edit images and videos</li> <li>• SMS, MMS, VCard, and broadcast message usage statistics on primary or secondary lines</li> <li>• Incoming and outgoing calls and voicemail usage statistics on primary or secondary lines</li> <li>• Emergency alerts are received or displayed statistics</li> <li>• Content searches within an app</li> <li>• Reading activity -- bookmarked, printed, or had the layout changed</li> </ul>
App or product state	<p>Information about Windows and application state, such as:</p> <ul style="list-style-type: none"> <li>• Start Menu and Taskbar pins</li> <li>• Online and offline status</li> <li>• App launch state -- with deep-links, such as Groove launching with an audio track to play or MMS launching to share a picture</li> <li>• Personalization impressions delivered</li> <li>• Whether the user clicked on, or hovered over, UI controls or hotspots</li> <li>• User provided feedback, such as Like, Dislike or a rating</li> <li>• Caret location or position within documents and media files -- how much has been read in a book in a single session, or how much of a song has been listened to.</li> </ul>
Purchasing	<p>Information about purchases made on the device, such as:</p> <ul style="list-style-type: none"> <li>• Product ID, edition ID and product URI</li> <li>• Offer details -- price</li> <li>• Date and time an order was requested</li> <li>• Microsoft Store client type -- web or native client</li> <li>• Purchase quantity and price</li> <li>• Payment type -- credit card type and PayPal</li> </ul>
Login properties	<p>Information about logins on the device, such as:</p> <ul style="list-style-type: none"> <li>• Login success or failure</li> <li>• Login sessions and state</li> </ul>



# Product and Service Performance data

This type of data includes details about the health of the device, operating system, apps, and drivers. Product and Service Performance data is equivalent to ISO/IEC 19944:2017 8.2.3.2.2 EUUI Telemetry data.

## Data Use for Product and Service Performance data

### For Diagnostics:

[Pseudonymized](#) Product and Service Performance data from Windows 10 is used by Microsoft to [provide](#) and [improve](#) Windows 10 and related Microsoft product and services. For example:

- Data about the reliability of content that appears in the [Windows Spotlight](#) (rotating lock screen images) is used for Windows Spotlight reliability investigations.
- Timing data about how quickly Cortana responds to voice commands is used to improve Cortana listening performance.
- Timing data about how quickly the facial recognition feature starts up and finishes is used to improve facial recognition performance.
- Data about when an Application Window fails to appear is used to investigate issues with Application Window reliability and performance.

### With (optional) Tailored experiences:

If a user has enabled Tailored experiences on the device, [pseudonymized](#) Product and Service Performance data from Windows 10 is used by Microsoft to [personalize](#), [recommend](#), and [offer](#) Microsoft products and services to Windows 10 users. Also, if a user has enabled Tailored experiences on the device, [pseudonymized](#) Product and Service Performance data from Windows 10 is used by Microsoft to [promote](#) third-party Windows apps, services, hardware, and peripherals to Windows 10 users.

- Data about battery performance on a device may be used to recommend settings changes that can improve battery performance.
- If data shows a device is running low on file storage, we may recommend Windows-compatible cloud storage solutions to free up space.
- If data shows the device is experiencing performance issues, we may provide recommendations for Windows apps that can help diagnose or resolve these issues. These may be free or paid apps.

**Microsoft doesn't use crash and hang dump data to [personalize](#), [recommend](#), [offer](#), or [promote](#) any product or service.**

## Data Description for Product and Service Performance data type

SUB-TYPE	DESCRIPTION AND EXAMPLES
----------	--------------------------

SUB-TYPE	DESCRIPTION AND EXAMPLES
Device health and crash data	<p>Information about the device and software health, such as:</p> <ul style="list-style-type: none"> <li>• Error codes and error messages, name and ID of the app, and process reporting the error</li> <li>• DLL library predicted to be the source of the error -- for example, xyz.dll</li> <li>• System generated files -- app or product logs and trace files to help diagnose a crash or hang</li> <li>• System settings, such as registry keys</li> <li>• User generated files -- files that are indicated as a potential cause for a crash or hang. For example, .doc, .ppt, .csv files</li> <li>• Details and counts of abnormal shutdowns, hangs, and crashes</li> <li>• Crash failure data -- operating system, operating system component, driver, device, and 1st and 3rd-party app data</li> <li>• Crash and hang dumps, including: <ul style="list-style-type: none"> <li>◦ The recorded state of the working memory at the point of the crash</li> <li>◦ Memory in-use by the kernel at the point of the crash.</li> <li>◦ Memory in-use by the application at the point of the crash</li> <li>◦ All the physical memory used by Windows at the point of the crash</li> <li>◦ Class and function name within the module that failed.</li> </ul> </li> </ul>

SUB-TYPE	DESCRIPTION AND EXAMPLES
Device performance and reliability data	<p>Information about the device and software performance, such as:</p> <ul style="list-style-type: none"> <li>• User interface interaction durations -- Start menu display times, browser tab switch times, app launch and switch times, and Cortana and Search performance and reliability</li> <li>• Device on and off performance -- Device boot, shutdown, power on and off, lock and unlock times, and user authentication times (fingerprint and face recognition durations)</li> <li>• In-app responsiveness -- time to set alarm, time to fully render in-app navigation menus, time to sync reading list, time to start GPS navigation, time to attach picture MMS, and time to complete a Microsoft Store transaction</li> <li>• User input responsiveness -- onscreen keyboard invocation times for different languages, time to show auto-complete words, pen or touch latencies, latency for handwriting recognition to words, Narrator screen reader responsiveness, and CPU score</li> <li>• UI and media performance and glitches versus smoothness -- video playback frame rate, audio glitches, animation glitches (stutter when bringing up Start), graphics score, time to first frame, play/pause/stop/seek responsiveness, time to render PDF, dynamic streaming of video from OneDrive performance</li> <li>• Disk footprint -- Free disk space, out of memory conditions, and disk score</li> <li>• Excessive resource utilization -- components impacting performance or battery life through high CPU usage during different screen and power states</li> <li>• Background task performance -- download times, Windows Update scan duration, Windows Defender Antivirus scan times, disk defrag times, mail fetch times, service startup and state transition times, and time to index on-device files for search results</li> <li>• Peripheral and devices -- USB device connection times, time to connect to a wireless display, printing times, network availability and connection times (time to connect to Wi-Fi, time to get an IP address from DHCP etc.), smart card authentication times, automatic brightness, and environmental response times</li> <li>• Device setup -- first setup experience times (time to install updates, install apps, connect to network, and so on), time to recognize connected devices (printer and monitor), and time to set up a Microsoft Account</li> <li>• Power and Battery life -- power draw by component (Process/CPU/GPU/Display), hours of time the screen is off, sleep state transition details, temperature and thermal throttling, battery drain in a power state (screen off or screen on), processes and components requesting power use while the screen is off, auto-brightness details, time device is plugged into AC versus battery, and battery state transitions</li> <li>• Service responsiveness -- Service URI, operation, latency, service success and error codes, and protocol</li> <li>• Diagnostic heartbeat -- regular signal used to validate the health of the diagnostics system</li> </ul>

SUB-TYPE	DESCRIPTION AND EXAMPLES
Movies	<p>Information about movie consumption functionality on the device. This isn't intended to capture user viewing, listening, or habits.</p> <ul style="list-style-type: none"> <li>• Video Width, height, color palette, encoding (compression) type, and encryption type</li> <li>• Instructions about how to stream content for the user -- the smooth streaming manifest of content file chunks that must be pieced together to stream the content based on screen resolution and bandwidth</li> <li>• URL for a specific two-second chunk of content if there is an error</li> <li>• Full-screen viewing mode details</li> </ul>
Music & TV	<p>Information about music and TV consumption on the device. This isn't intended to capture user viewing, listening, or habits.</p> <ul style="list-style-type: none"> <li>• Service URL for song being downloaded from the music service -- collected when an error occurs to facilitate restoration of service</li> <li>• Content type (video, audio, or surround audio)</li> <li>• Local media library collection statistics -- number of purchased tracks and number of playlists</li> <li>• Region mismatch -- User's operating system region and Xbox Live region</li> </ul>
Reading	<p>Information about reading consumption functionality on the device. This isn't intended to capture user viewing, listening, or habits.</p> <ul style="list-style-type: none"> <li>• App accessing content and status and options used to open a Microsoft Store book</li> <li>• Language of the book</li> <li>• Time spent reading content</li> <li>• Content type and size details</li> </ul>
Photos App	<p>Information about photos usage on the device. This isn't intended to capture user viewing, listening, or habits.</p> <ul style="list-style-type: none"> <li>• File source data -- local, SD card, network device, and OneDrive</li> <li>• Image and video resolution, video length, file sizes types, and encoding</li> <li>• Collection view or full screen viewer use and duration of view</li> </ul>
On-device file query	<p>Information about local search activity on the device, such as:</p> <ul style="list-style-type: none"> <li>• Kind of query issued and index type (ConstraintIndex or SystemIndex)</li> <li>• Number of items requested and retrieved</li> <li>• File extension of search result with which the user interacted</li> <li>• Launched item type, file extension, index of origin, and the App ID of the opening app</li> <li>• Name of process calling the indexer and the amount of time to service the query</li> <li>• A hash of the search scope (file, Outlook, OneNote, or IE history). The state of the indices (fully optimized, partially optimized, or being built)</li> </ul>

SUB-TYPE	DESCRIPTION AND EXAMPLES
Entitlements	Information about entitlements on the device, such as: <ul style="list-style-type: none"> <li>• Service subscription status and errors</li> <li>• DRM and license rights details -- Groove subscription or operating system volume license</li> <li>• Entitlement ID, lease ID, and package ID of the install package</li> <li>• Entitlement revocation</li> <li>• License type (trial, offline versus online) and duration</li> <li>• License usage session</li> </ul>

## Software Setup and Inventory data

This type of data includes software installation and update information on the device. Software Setup and Inventory Data is a sub-type of ISO/IEC 19944:2017 8.2.3.2.4 Observed Usage of the Service Capability.

### Data Use for Software Setup and Inventory data

#### For Diagnostics:

[Pseudonymized](#) Software Setup and Inventory data from Windows 10 is used by Microsoft to [provide](#) and [improve](#) Windows 10 and related Microsoft product and services. For example:

- Data about the specific drivers that are installed on a device is used to understand whether there are any hardware or driver compatibility issues which should block or delay a Windows update.
- Data about when a download starts and finishes on a device is used to understand and address download problems.
- Data about the specific Microsoft Store apps that are installed on a device is used to determine which app updates to provide to the device.
- Data about the antimalware installed on a device is used to understand malware transmissions vectors.

#### With (optional) Tailored experiences:

If a user has enabled Tailored experiences on the device, [pseudonymized](#) Software Setup and Inventory data from Windows 10 is used by Microsoft to [personalize](#), [recommend](#), and [offer](#) Microsoft products and services to Windows 10 users. Also, if a user has enabled Tailored experiences on the device, [pseudonymized](#) Software Setup and Inventory data from Windows 10 is used by Microsoft to [promote](#) third-party Windows apps, services, hardware, and peripherals to Windows 10 users. For example:

- Data about the specific apps that are installed on a device is used to provide recommendations for similar or complementary apps in the Microsoft Store.

### Data Description for Software Setup and Inventory data type

SUB-TYPE	DESCRIPTION AND EXAMPLES
----------	--------------------------

SUB-TYPE	DESCRIPTION AND EXAMPLES
Installed Applications and Install History	<p>Information about apps, drivers, update packages, or operating system components installed on the device, such as:</p> <ul style="list-style-type: none"> <li>• App, driver, update package, or component's Name, ID, or Package Family Name</li> <li>• Product, SKU, availability, catalog, content, and Bundle IDs</li> <li>• Operating system component, app or driver publisher, language, version and type (Win32 or UWP)</li> <li>• Install date, method, install directory, and count of install attempts</li> <li>• MSI package and product code</li> <li>• Original operating system version at install time</li> <li>• User, administrator, or mandatory installation or update</li> <li>• Installation type -- clean install, repair, restore, OEM, retail, upgrade, or update</li> </ul>
Device update information	<p>Information about Windows Update, such as:</p> <ul style="list-style-type: none"> <li>• Update Readiness analysis of device hardware, operating system components, apps, and drivers (progress, status, and results)</li> <li>• Number of applicable updates, importance, and type</li> <li>• Update download size and source -- CDN or LAN peers</li> <li>• Delay upgrade status and configuration</li> <li>• Operating system uninstall and rollback status and count</li> <li>• Windows Update server and service URL</li> <li>• Windows Update machine ID</li> <li>• Windows Insider build details</li> </ul>

## Browsing History data

This type of data includes details about web browsing in the Microsoft browsers. Browsing History data is equivalent to ISO/IEC 19944:2017 8.2.3.2.8 Client side browsing history.

### Data Use for Browsing History data

#### For Diagnostics:

[Pseudonymized](#) Browsing History data from Windows 10 is used by Microsoft to [provide](#) and [improve](#) Windows 10 and related Microsoft product and services. For example:

- Data about when the **Block Content** dialog box has been shown is used for investigations of blocked content.
- Data about potentially abusive or malicious domains is used to make updates to Microsoft Edge and Windows Defender SmartScreen to warn users about the domain.
- Data about when the **Address** bar is used for navigation purposes is used to improve the Suggested Sites feature and to understand and address problems arising from navigation.
- Data about when a Web Notes session starts is used to measure popular domains and URLs for the Web Notes feature.
- Data about when a default **Home** page is changed by a user is used to measure which default **Home** pages

are the most popular and how often users change the default **Home** page.

### With (optional) Tailored experiences:

If a user has enabled Tailored experiences on the device, [pseudonymized](#) Browsing History data from Windows 10 is used by Microsoft to [personalize](#), [recommend](#), and [offer](#) Microsoft products and services to Windows 10 users. Also, if a user has enabled Tailored experiences on the device, [pseudonymized](#) Browsing History data from Windows 10 is used by Microsoft to [promote](#) third-party Windows apps, services, hardware, and peripherals to Windows 10 users. For example:

- We may recommend that a user download a compatible app from the Microsoft Store if they have browsed to the related website. For example, if a user uses the Facebook website, we may recommend the Facebook app.

### Data Description for Browsing History data type

SUB-TYPE	DESCRIPTION AND EXAMPLES
Microsoft browser data	Information about <b>Address</b> bar and <b>Search</b> box performance on the device, such as: <ul style="list-style-type: none"><li>• Text typed in <b>Address</b> bar and <b>Search</b> box</li><li>• Text selected for an <b>Ask Cortana</b> search</li><li>• Service response time</li><li>• Auto-completed text, if there was an auto-complete</li><li>• Navigation suggestions provided based on local history and favorites</li><li>• Browser ID</li><li>• URLs (may include search terms)</li><li>• Page title</li></ul>

## Inking Typing and Speech Utterance data

This type of data gathers details about the voice, inking, and typing input features on the device. Inking, Typing and Speech Utterance data is a sub-type of ISO/IEC 19944:2017 8.2.3.2.1 End User Identifiable information.

### Data Use for Inking, Typing, and Speech Utterance data

#### For Diagnostics:

[Anonymized](#) Inking, Typing, and Speech Utterance data from Windows 10 is used by Microsoft to [improve](#) natural language capabilities in Microsoft products and services. For example:

- Data about words marked as spelling mistakes and replaced with another word from the context menu is used to improve the spelling feature.
- Data about alternate words shown and selected by the user after right-clicking is used to improve the word recommendation feature.
- Data about auto-corrected words that were restored back to the original word by the user is used to improve the auto-correct feature.
- Data about whether Narrator detected and recognized a touch gesture is used to improve touch gesture recognition.
- Data about handwriting samples sent from the Handwriting Panel is used to help Microsoft improve handwriting recognition.

### With (optional) Tailored experiences:

**Microsoft doesn't use Windows Inking, Typing, and Speech Utterance data for Tailored experiences.**

### Data Description for Inking, Typing, and Speech Utterance data type

SUB-TYPE	DESCRIPTION AND EXAMPLES
Voice, inking, and typing	<p>Information about voice, inking and typing features, such as:</p> <ul style="list-style-type: none"> <li>• Type of pen used (highlighter, ball point, or pencil), pen color, stroke height and width, and how long it is used</li> <li>• Pen gestures (click, double click, pan, zoom, or rotate)</li> <li>• Palm Touch x,y coordinates</li> <li>• Input latency, missed pen signals, number of frames, strokes, first frame commit time, and sample rate</li> <li>• Ink strokes written, text before and after the ink insertion point, recognized text entered, input language -- processed to remove identifiers, sequencing information, and other data (such as email addresses and numeric values), which could be used to reconstruct the original content or associate the input to the user</li> <li>• Text input from Windows 10 Mobile on-screen keyboards, except from password fields and private sessions -- processed to remove identifiers, sequencing information, and other data (such as email addresses and numeric values), which could be used to reconstruct the original content or associate the input to the user</li> <li>• Text of speech recognition results -- result codes and recognized text</li> <li>• Language and model of the recognizer and the System Speech language</li> <li>• App ID using speech features</li> <li>• Whether user is known to be a child</li> <li>• Confidence and success or failure of speech recognition</li> </ul>

## ISO/IEC 19944:2017-specific terminology

This table provides the ISO/IEC 19944:2017-specific definitions for use and de-identification qualifiers used in this article.

TERM	ISO/IEC 19944:2017 REFERENCE	MICROSOFT USAGE NOTES
--	9.3.2 Provide	Use of a specified data category by a Microsoft product or service to protect and provide the described service, including, (i) troubleshoot and fix issues with the product or service or (ii) provide product or service updates.
--	9.3.3 Improve	Use of a specified data category to improve or increase the quality of a Microsoft product or service. Those improvements may be available to end users.
--	9.3.4 Personalize	Use of the specified data categories to create a customized experience for the end user in any Microsoft product or service.



TERM	ISO/IEC 19944:2017 REFERENCE	MICROSOFT USAGE NOTES
<a href="#">---</a>	9.3.4 Personalize	<p>“Recommend” means use of the specified data categories to Personalize (9.3.4) the end user’s experience by recommending Microsoft products or services that can be accessed without the need to make a purchase or pay money.</p> <p>Use of the specified data categories give recommendations about Microsoft products or services the end user may act on where the recommendation is (i) contextually relevant to the product or service in which it appears, (ii) that can be accessed without the need to make a purchase or pay money, and (iii) Microsoft receives no compensation for the placement.</p>
<a href="#">---</a>	9.3.5 Offer upgrades or upsell	<p>Implies the source of the data is Microsoft products and services, and the upgrades offered come from Microsoft products and services that are relevant to the context of the current capability. The target audience for the offer is Microsoft customers.</p> <p>Specifically, use of the specified data categories to make an offer or upsell new capability or capacity of a Microsoft product or service which is (i) contextually relevant to the product or service in which it appears; (ii) likely to result in additional future revenue for Microsoft from end user; and (iii) Microsoft receives no consideration for placement.</p>
<a href="#">---</a>	9.3.6 Market/advertise/promote	Use of the specified data categories to promote a product or service in or on a first-party Microsoft product or service.

DATA IDENTIFICATION QUALIFIERS	ISO/IEC 19944:2017 REFERENCE	MICROSOFT USAGE NOTES
<a href="#">-----</a>	8.3.3 Pseudonymized data	As defined
<a href="#">-----</a>	8.3.5 Anonymized data	As defined
<a href="#">-----</a>	8.3.6 Aggregated data	As defined

# Windows 10 diagnostic data for the Full diagnostic data level

7/9/2018 • 13 minutes to read • [Edit Online](#)

## Applies to:

- Windows 10, version 1703

Microsoft collects Windows diagnostic data to keep Windows up-to-date, secure, and operating properly. It also helps us improve Windows and, for users who have turned on “tailored experiences”, can be used to provide more relevant tips and recommendations to tailor Microsoft products to the user’s needs. This article describes all types diagnostic data collected by Windows at the Full diagnostic data level (inclusive of data collected at Basic), with comprehensive examples of data we collect per each type. For additional, detailed technical descriptions of Basic data items, see [Windows 10, version 1709 Basic level diagnostic events and fields](#) and [Windows 10, version 1703 Basic level diagnostic events and fields](#).

The data covered in this article is grouped into the following categories:

- Common Data (diagnostic header information)
- Device, Connectivity, and Configuration data
- Product and Service Usage data
- Product and Service Performance data
- Software Setup and Inventory data
- Browsing History data
- Inking, Typing, and Speech Utterance data

### NOTE

The majority of diagnostic data falls into the first four categories.

## Common data

Most diagnostic events contain a header of common data:

CATEGORY NAME	EXAMPLES
---------------	----------

CATEGORY NAME	EXAMPLES
Common Data	<p>Information that is added to most diagnostic events, if relevant and available:</p> <ul style="list-style-type: none"> <li>• OS name, version, build, and <a href="#">locale</a></li> <li>• User ID -- a unique identifier associated with the user's Microsoft Account (if one is used) or local account. The user's Microsoft Account identifier is not collected from devices configured to send Basic diagnostic data</li> <li>• Xbox UserID</li> <li>• Environment from which the event was logged -- Application ID of app or component that logged the event, Session GUID. Used to track events over a given period of time such the period an app is running or between boots of the OS.</li> <li>• The diagnostic event name, Event ID, <a href="#">ETW</a> opcode, version, schema signature, keywords, and flags</li> <li>• HTTP header information, including the IP address. This IP address is the source address that's provided by the network packet header and received by the diagnostics ingestion service.</li> <li>• Various IDs that are used to correlate and sequence related events together.</li> <li>• Device ID. This is not the user provided device name, but an ID that is unique for that device.</li> <li>• Device class -- Desktop, Server, or Mobile</li> <li>• Event collection time</li> <li>• Diagnostic level -- Basic or Full, Sample level -- for sampled data, what sample level is this device opted into</li> </ul>

## Device, Connectivity, and Configuration data

This type of data includes details about the device, its configuration and connectivity capabilities, and status.

CATEGORY NAME	EXAMPLES
Device properties	<p>Information about the OS and device hardware, such as:</p> <ul style="list-style-type: none"> <li>• OS - version name, Edition</li> <li>• Installation type, subscription status, and genuine OS status</li> <li>• Processor architecture, speed, number of cores, manufacturer, and model</li> <li>• OEM details --manufacturer, model, and serial number</li> <li>• Device identifier and Xbox serial number</li> <li>• Firmware/BIOS -- type, manufacturer, model, and version</li> <li>• Memory -- total memory, video memory, speed, and how much memory is available after the device has reserved memory</li> <li>• Storage -- total capacity and disk type</li> <li>• Battery -- charge capacity and InstantOn support</li> <li>• Hardware chassis type, color, and form factor</li> <li>• Is this a virtual machine?</li> </ul>

CATEGORY NAME	EXAMPLES
Device capabilities	<p>Information about the specific device capabilities such as:</p> <ul style="list-style-type: none"> <li>• Camera -- whether the device has a front facing, a rear facing camera, or both.</li> <li>• Touch screen -- does the device include a touch screen? If so, how many hardware touch points are supported?</li> <li>• Processor capabilities -- CompareExchange128, LahfSahf, NX, PrefetchW, and SSE2</li> <li>• Trusted Platform Module (TPM) – whether present and what version</li> <li>• Virtualization hardware -- whether an IOMMU is present, SLAT support, is virtualization enabled in the firmware</li> <li>• Voice – whether voice interaction is supported and the number of active microphones</li> <li>• Number of displays, resolutions, DPI</li> <li>• Wireless capabilities</li> <li>• OEM or platform face detection</li> <li>• OEM or platform video stabilization and quality level set</li> <li>• Advanced Camera Capture mode (HDR vs. LowLight), OEM vs. platform implementation, HDR probability, and Low Light probability</li> </ul>
Device preferences and settings	<p>Information about the device settings and user preferences such as:</p> <ul style="list-style-type: none"> <li>• User Settings – System, Device, Network &amp; Internet, Personalization, Cortana, Apps, Accounts, Time &amp; Language, Gaming, Ease of Access, Privacy, Update &amp; Security</li> <li>• User-provided device name</li> <li>• Whether device is domain-joined, or cloud-domain joined (i.e. part of a company-managed network)</li> <li>• Hashed representation of the domain name</li> <li>• MDM (mobile device management) enrollment settings and status</li> <li>• BitLocker, Secure Boot, encryption settings, and status</li> <li>• Windows Update settings and status</li> <li>• Developer Unlock settings and status</li> <li>• Default app choices</li> <li>• Default browser choice</li> <li>• Default language settings for app, input, keyboard, speech, and display</li> <li>• App store update settings</li> <li>• Enterprise OrganizationID, Commercial ID</li> </ul>
Device peripherals	<p>Information about the device peripherals such as:</p> <ul style="list-style-type: none"> <li>• Peripheral name, device model, class, manufacturer and description</li> <li>• Peripheral device state, install state, and checksum</li> <li>• Driver name, package name, version, and manufacturer</li> <li>• HWID - A hardware vendor defined ID to match a device to a driver <a href="#">INF file</a></li> <li>• Driver state, problem code, and checksum</li> <li>• Whether driver is kernel mode, signed, and image size</li> </ul>

CATEGORY NAME	EXAMPLES
Device network info	<p>Information about the device network configuration such as:</p> <ul style="list-style-type: none"> <li>• Network system capabilities</li> <li>• Local or Internet connectivity status</li> <li>• Proxy, gateway, DHCP, DNS details and addresses</li> <li>• Paid or free network</li> <li>• Wireless driver is emulated or not</li> <li>• Access point mode capable</li> <li>• Access point manufacturer, model, and MAC address</li> <li>• WDI Version</li> <li>• Name of networking driver service</li> <li>• Wi-Fi Direct details</li> <li>• Wi-Fi device hardware ID and manufacturer</li> <li>• Wi-Fi scan attempt counts and item counts</li> <li>• Mac randomization is supported/enabled or not</li> <li>• Number of spatial streams and channel frequencies supported</li> <li>• Manual or Auto Connect enabled</li> <li>• Time and result of each connection attempt</li> <li>• Airplane mode status and attempts</li> <li>• Interface description provided by the manufacturer</li> <li>• Data transfer rates</li> <li>• Cipher algorithm</li> <li>• Mobile Equipment ID (IMEI) and Mobile Country Code (MCCO)</li> <li>• Mobile operator and service provider name</li> <li>• Available SSIDs and BSSIDs</li> <li>• IP Address type -- IPv4 or IPv6</li> <li>• Signal Quality percentage and changes</li> <li>• Hotspot presence detection and success rate</li> <li>• TCP connection performance</li> <li>• Miracast device names</li> <li>• Hashed IP address</li> </ul>

## Product and Service Usage data

This type of data includes details about the usage of the device, operating system, applications and services.

CATEGORY NAME	EXAMPLES
---------------	----------

CATEGORY NAME	EXAMPLES
App usage	<p>Information about Windows and application usage such as:</p> <ul style="list-style-type: none"> <li>• OS component and app feature usage</li> <li>• User navigation and interaction with app and Windows features. This could potentially include user input, such as name of a new alarm set, user menu choices, or user favorites.</li> <li>• Time of and count of app/component launches, duration of use, session GUID, and process ID</li> <li>• App time in various states – running foreground or background, sleeping, or receiving active user interaction</li> <li>• User interaction method and duration – whether and length of time user used the keyboard, mouse, pen, touch, speech, or game controller</li> <li>• Cortana launch entry point/reason</li> <li>• Notification delivery requests and status</li> <li>• Apps used to edit images and videos</li> <li>• SMS, MMS, VCard, and broadcast message usage statistics on primary or secondary line</li> <li>• Incoming and Outgoing calls and Voicemail usage statistics on primary or secondary line</li> <li>• Emergency alerts are received or displayed statistics</li> <li>• Content searches within an app</li> <li>• Reading activity -- bookmarking used, print used, layout changed</li> </ul>
App or product state	<p>Information about Windows and application state such as:</p> <ul style="list-style-type: none"> <li>• Start Menu and Taskbar pins</li> <li>• Online/Offline status</li> <li>• App launch state -- with deep-link such as Groove launched with an audio track to play, or share contract such as MMS launched to share a picture.</li> <li>• Personalization impressions delivered</li> <li>• Whether the user clicked or hovered on UI controls or hotspots</li> <li>• User feedback Like or Dislike or rating was provided</li> <li>• Caret location or position within documents and media files -- how much of a book has been read in a single session or how much of a song has been listened to.</li> </ul>
Login properties	<ul style="list-style-type: none"> <li>• Login success or failure</li> <li>• Login sessions and state</li> </ul>

## Product and Service Performance data

This type of data includes details about the health of the device, operating system, apps and drivers.

CATEGORY NAME	DESCRIPTION AND EXAMPLES
---------------	--------------------------

CATEGORY NAME	DESCRIPTION AND EXAMPLES
Device health and crash data	<p>Information about the device and software health such as:</p> <ul style="list-style-type: none"><li>• Error codes and error messages, name and ID of the app, and process reporting the error</li><li>• DLL library predicted to be the source of the error -- xyz.dll</li><li>• System generated files -- app or product logs and trace files to help diagnose a crash or hang</li><li>• System settings such as registry keys</li><li>• User generated files – .doc, .ppt, .csv files where they are indicated as a potential cause for a crash or hang</li><li>• Details and counts of abnormal shutdowns, hangs, and crashes</li><li>• Crash failure data – OS, OS component, driver, device, 1st and 3rd party app data</li><li>• Crash and Hang dumps<ul style="list-style-type: none"><li>◦ The recorded state of the working memory at the point of the crash.</li><li>◦ Memory in use by the kernel at the point of the crash.</li><li>◦ Memory in use by the application at the point of the crash.</li><li>◦ All the physical memory used by Windows at the point of the crash.</li><li>◦ Class and function name within the module that failed.</li></ul></li></ul>

CATEGORY NAME	DESCRIPTION AND EXAMPLES
Device performance and reliability data	<p>Information about the device and software performance such as:</p> <ul style="list-style-type: none"> <li>• User Interface interaction durations -- Start Menu display times, browser tab switch times, app launch and switch times, and Cortana and search performance and reliability.</li> <li>• Device on/off performance -- Device boot, shutdown, power on/off, lock/unlock times, and user authentication times (fingerprint and face recognition durations).</li> <li>• In-app responsiveness -- time to set alarm, time to fully render in-app navigation menus, time to sync reading list, time to start GPS navigation, time to attach picture MMS, and time to complete a Microsoft Store transaction.</li> <li>• User input responsiveness – onscreen keyboard invocation times for different languages, time to show auto-complete words, pen or touch latencies, latency for handwriting recognition to words, Narrator screen reader responsiveness, and CPU score.</li> <li>• UI and media performance and glitches/smoothness -- video playback frame rate, audio glitches, animation glitches (stutter when bringing up Start), graphics score, time to first frame, play/pause/stop/seek responsiveness, time to render PDF, dynamic streaming of video from OneDrive performance</li> <li>• Disk footprint -- Free disk space, out of memory conditions, and disk score.</li> <li>• Excessive resource utilization – components impacting performance or battery life through high CPU usage during different screen and power states</li> <li>• Background task performance -- download times, Windows Update scan duration, Windows Defender Antivirus scan times, disk defrag times, mail fetch times, service startup and state transition times, and time to index on-device files for search results</li> <li>• Peripheral and devices -- USB device connection times, time to connect to a wireless display, printing times, network availability and connection times (time to connect to Wi-Fi, time to get an IP address from DHCP etc.), smart card authentication times, automatic brightness environmental response times</li> <li>• Device setup -- first setup experience times (time to install updates, install apps, connect to network etc.), time to recognize connected devices (printer and monitor), and time to setup Microsoft Account.</li> <li>• Power and Battery life – power draw by component (Process/CPU/GPU/Display), hours of screen off time, sleep state transition details, temperature and thermal throttling, battery drain in a power state (screen off or screen on), processes and components requesting power use during screen off, auto-brightness details, time device is plugged into AC vs. battery, battery state transitions</li> <li>• Service responsiveness - Service URI, operation, latency, service success/error codes, and protocol.</li> <li>• Diagnostic heartbeat – regular signal to validate the health of the diagnostics system</li> </ul>



CATEGORY NAME	DESCRIPTION AND EXAMPLES
Movies	<p>Information about movie consumption functionality on the device. This isn't intended to capture user viewing, listening or habits.</p> <ul style="list-style-type: none"> <li>• Video Width, height, color pallet, encoding (compression) type, and encryption type</li> <li>• Instructions for how to stream content for the user -- the smooth streaming manifest of chunks of content files that must be pieced together to stream the content based on screen resolution and bandwidth</li> <li>• URL for a specific two second chunk of content if there is an error</li> <li>• Full screen viewing mode details</li> </ul>
Music & TV	<p>Information about music and TV consumption on the device. This isn't intended to capture user viewing, listening or habits.</p> <ul style="list-style-type: none"> <li>• Service URL for song being downloaded from the music service – collected when an error occurs to facilitate restoration of service</li> <li>• Content type (video, audio, surround audio)</li> <li>• Local media library collection statistics -- number of purchased tracks, number of playlists</li> <li>• Region mismatch -- User OS Region, and Xbox Live region</li> </ul>
Reading	<p>Information about reading consumption functionality on the device. This isn't intended to capture user viewing, listening or habits.</p> <ul style="list-style-type: none"> <li>• App accessing content and status and options used to open a Microsoft Store book</li> <li>• Language of the book</li> <li>• Time spent reading content</li> <li>• Content type and size details</li> </ul>
Photos App	<p>Information about photos usage on the device. This isn't intended to capture user viewing, listening or habits.</p> <ul style="list-style-type: none"> <li>• File source data -- local, SD card, network device, and OneDrive</li> <li>• Image &amp; video resolution, video length, file sizes types and encoding</li> <li>• Collection view or full screen viewer use and duration of view</li> </ul>

CATEGORY NAME	DESCRIPTION AND EXAMPLES
On-device file query	<p>Information about local search activity on the device such as:</p> <ul style="list-style-type: none"> <li>• Kind of query issued and index type (ConstraintIndex, SystemIndex)</li> <li>• Number of items requested and retrieved</li> <li>• File extension of search result user interacted with</li> <li>• Launched item kind, file extension, index of origin, and the App ID of the opening app.</li> <li>• Name of process calling the indexer and time to service the query.</li> <li>• A hash of the search scope (file, Outlook, OneNote, IE history)</li> <li>• The state of the indices (fully optimized, partially optimized, being built)</li> </ul>
Purchasing	<p>Information about purchases made on the device such as:</p> <ul style="list-style-type: none"> <li>• Product ID, edition ID and product URI</li> <li>• Offer details -- price</li> <li>• Order requested date/time</li> <li>• Store client type -- web or native client</li> <li>• Purchase quantity and price</li> <li>• Payment type -- credit card type and PayPal</li> </ul>
Entitlements	<p>Information about entitlements on the device such as:</p> <ul style="list-style-type: none"> <li>• Service subscription status and errors</li> <li>• DRM and license rights details -- Groove subscription or OS volume license</li> <li>• Entitlement ID, lease ID, and package ID of the install package</li> <li>• Entitlement revocation</li> <li>• License type (trial, offline vs online) and duration</li> <li>• License usage session</li> </ul>

## Software Setup and Inventory data

This type of data includes software installation and update information on the device.

CATEGORY NAME	DATA EXAMPLES
Installed Applications and Install History	<p>Information about apps, drivers, update packages, or OS components installed on the device such as:</p> <ul style="list-style-type: none"> <li>• App, driver, update package, or component's Name, ID, or Package Family Name</li> <li>• Product, SKU, availability, catalog, content, and Bundle IDs</li> <li>• OS component, app or driver publisher, language, version and type (Win32 or UWP)</li> <li>• Install date, method, and install directory, count of install attempts</li> <li>• MSI package code and product code</li> <li>• Original OS version at install time</li> <li>• User or administrator or mandatory installation/update</li> <li>• Installation type – clean install, repair, restore, OEM, retail, upgrade, and update</li> </ul>

CATEGORY NAME	DATA EXAMPLES
Device update information	Information about Windows Update such as: <ul style="list-style-type: none"> <li>• Update Readiness analysis of device hardware, OS components, apps, and drivers (progress, status, and results)</li> <li>• Number of applicable updates, importance, type</li> <li>• Update download size and source -- CDN or LAN peers</li> <li>• Delay upgrade status and configuration</li> <li>• OS uninstall and rollback status and count</li> <li>• Windows Update server and service URL</li> <li>• Windows Update machine ID</li> <li>• Windows Insider build details</li> </ul>

## Browsing History data

This type of data includes details about web browsing in the Microsoft browsers.

CATEGORY NAME	DESCRIPTION AND EXAMPLES
Microsoft browser data	Information about Address bar and search box performance on the device such as: <ul style="list-style-type: none"> <li>• Text typed in address bar and search box</li> <li>• Text selected for Ask Cortana search</li> <li>• Service response time</li> <li>• Auto-completed text if there was an auto-complete</li> <li>• Navigation suggestions provided based on local history and favorites</li> <li>• Browser ID</li> <li>• URLs (which may include search terms)</li> <li>• Page title</li> </ul>

## Inking Typing and Speech Utterance data

This type of data gathers details about the voice, inking, and typing input features on the device.

CATEGORY NAME	DESCRIPTION AND EXAMPLES
---------------	--------------------------

CATEGORY NAME	DESCRIPTION AND EXAMPLES
Voice, inking, and typing	<p>Information about voice, inking and typing features such as:</p> <ul style="list-style-type: none"><li>• Type of pen used (highlighter, ball point, pencil), pen color, stroke height and width, and how long it is used</li><li>• Pen gestures (click, double click, pan, zoom, rotate)</li><li>• Palm Touch x,y coordinates</li><li>• Input latency, missed pen signals, number of frames, strokes, first frame commit time, sample rate</li><li>• Ink strokes written, text before and after the ink insertion point, recognized text entered, Input language - processed to remove identifiers, sequencing information, and other data (such as email addresses and numeric values) which could be used to reconstruct the original content or associate the input to the user.</li><li>• Text input from Windows Mobile on-screen keyboards except from password fields and private sessions - processed to remove identifiers, sequencing information, and other data (such as email addresses, and numeric values) which could be used to reconstruct the original content or associate the input to the user.</li><li>• Text of speech recognition results -- result codes and recognized text</li><li>• Language and model of the recognizer, System Speech language</li><li>• App ID using speech features</li><li>• Whether user is known to be a child</li><li>• Confidence and Success/Failure of speech recognition</li></ul>

# Manage Windows 10 connection endpoints

7/2/2018 • 18 minutes to read • [Edit Online](#)

## Applies to

- Windows 10, version 1709 and later

Some Windows components, app, and related services transfer data to Microsoft network endpoints. Some examples include:

- Connecting to Microsoft Office and Windows sites to download the latest app and security updates.
- Connecting to email servers to send and receive email.
- Connecting to the web for every day web browsing.
- Connecting to the cloud to store and access backups.
- Using your location to show a weather forecast.

This article lists different endpoints that are available on a clean installation of Windows 10, version 1709 and later. Details about the different ways to control traffic to these endpoints are covered in [Manage connections from Windows operating system components to Microsoft services](#). Where applicable, each endpoint covered in this topic includes a link to specific details about how to control traffic to it.

We used the following methodology to derive these network endpoints:

1. Set up the latest version of Windows 10 on a test virtual machine using the default settings.
2. Leave the devices running idle for a week (that is, a user is not interacting with the system/device).
3. Use globally accepted network protocol analyzer/capturing tools and log all background egress traffic.
4. Compile reports on traffic going to public IP addresses.
5. The test virtual machine was logged in using a local account and was not joined to a domain or Azure Active Directory.

### NOTE

Microsoft uses global load balancers that can appear in network trace-routes. For example, an endpoint for \*.akadns.net might be used to load balance requests to an Azure datacenter, which can change over time.

## Windows 10 Enterprise connection endpoints

### Apps

The following endpoint is used to download updates to the Weather app Live Tile. If you [turn off traffic to this endpoint](#), no Live Tiles will be updated.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
explorer	HTTP	tile-service.weather.microsoft.com	1709
	HTTP	blob.weather.microsoft.com	1803

The following endpoint is used for OneNote Live Tile. To turn off traffic for this endpoint, either uninstall OneNote or [disable the Microsoft Store](#). If you disable the Microsoft store, other Store apps cannot be installed or updated. Additionally, the Microsoft Store won't be able to revoke malicious Store apps and users will still be able to open them.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
	HTTPS	cdn.onenote.net/livetile/? Language=en-US	1709

The following endpoints are used for Twitter updates. To turn off traffic for these endpoints, either uninstall Twitter or [disable the Microsoft Store](#). If you disable the Microsoft store, other Store apps cannot be installed or updated. Additionally, the Microsoft Store won't be able to revoke malicious Store apps and users will still be able to open them.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
	HTTPS	wildcard.twimg.com	1709
svchost.exe		oem.twimg.com/windows/tile.xml	1709

The following endpoint is used for Facebook updates. To turn off traffic for this endpoint, either uninstall Facebook or [disable the Microsoft Store](#). If you disable the Microsoft store, other Store apps cannot be installed or updated. Additionally, the Microsoft Store won't be able to revoke malicious Store apps and users will still be able to open them.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
		star-mini.c10r.facebook.com	1709

The following endpoint is used by the Photos app to download configuration files, and to connect to the Office 365 portal's shared infrastructure, including Office Online. To turn off traffic for this endpoint, either uninstall the Photos app or [disable the Microsoft Store](#). If you disable the Microsoft store, other Store apps cannot be installed or updated. Additionally, the Microsoft Store won't be able to revoke malicious Store apps and users will still be able to open them.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
WindowsApps\Microsoft.Windows.Photos	HTTPS	evoke-windowservices-tas.msedge.net	1709

The following endpoint is used for Candy Crush Saga updates. To turn off traffic for this endpoint, either uninstall Candy Crush Saga or [disable the Microsoft Store](#). If you disable the Microsoft store, other Store apps cannot be installed or updated. Additionally, the Microsoft Store won't be able to revoke malicious Store apps and users will still be able to open them.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
	TLS v1.2	candycrushsoda.king.com	1709

The following endpoint is used for by the Microsoft Wallet app. To turn off traffic for this endpoint, either uninstall the Wallet app or [disable the Microsoft Store](#). If you disable the Microsoft store, other Store apps cannot be installed or updated. Additionally, the Microsoft Store won't be able to revoke malicious Store apps and users will still be able to open them.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
system32\AppHostRegistrationVerifier.exe	HTTPS	wallet.microsoft.com	1709

The following endpoint is used by the Groove Music app for update HTTP handler status. If you [turn off traffic for this endpoint](#), apps for websites won't work and customers who visit websites (such as [mediaredirect.microsoft.com](#)) that are registered with their associated app (such as Groove Music) will stay at the website and won't be able to directly launch the app.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
system32\AppHostRegistrationVerifier.exe	HTTPS	mediaredirect.microsoft.com	1709

## Cortana and Search

The following endpoint is used to get images that are used for Microsoft Store suggestions. If you [turn off traffic for this endpoint](#), you will block images that are used for Microsoft Store suggestions.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
searchui	HTTPS	store-images.s-microsoft.com	1709

The following endpoint is used to update Cortana greetings, tips, and Live Tiles. If you [turn off traffic for this endpoint](#), you will block updates to Cortana greetings, tips, and Live Tiles.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
backgroundtaskhost	HTTPS	<a href="http://www.bing.com/client">www.bing.com/client</a>	1709

The following endpoint is used to configure parameters, such as how often the Live Tile is updated. It's also used to activate experiments. If you [turn off traffic for this endpoint](#), parameters would not be updated and the device would no longer participate in experiments.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
backgroundtaskhost	HTTPS	<a href="http://www.bing.com/proactive">www.bing.com/proactive</a>	1709

The following endpoint is used by Cortana to report diagnostic and diagnostic data information. If you [turn off traffic for this endpoint](#), Microsoft won't be aware of issues with Cortana and won't be able to fix them.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
searchui backgroundtaskhost	HTTPS	<a href="http://www.bing.com/threshold/xls.aspx">www.bing.com/threshold/xls.aspx</a>	1709

## Certificates

The following endpoint is used by the Automatic Root Certificates Update component to automatically check the list of trusted authorities on Windows Update to see if an update is available. It is possible to [turn off traffic to this endpoint](#), but that is not recommended because when root certificates are updated over time, applications and websites may stop working because they did not receive an updated root certificate the application uses.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
svchost	HTTP	ctldl.windowsupdate.com	1709

The following endpoints are used to download certificates that are publicly known to be fraudulent. These settings are critical for both Windows security and the overall security of the Internet. We do not recommend blocking this endpoint. If traffic to this endpoint is turned off, Windows no longer automatically downloads certificates known to be fraudulent, which increases the attack vector on the device.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
svchost	HTTP	ctldl.windowsupdate.com	1709

## Device authentication

The following endpoint is used to authenticate a device. If you [turn off traffic for this endpoint](#), the device will not be authenticated.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
	HTTPS	login.live.com/ppsecure	1709

## Device metadata

The following endpoint is used to retrieve device metadata. If you [turn off traffic for this endpoint](#), metadata will not be updated for the device.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
		dmd.metaservices.microsoft.com.akadns.net	1709
	HTTP	dmd.metaservices.microsoft.com	1803

## Diagnostic Data



The following endpoint is used by the Connected User Experiences and Telemetry component and connects to the Microsoft Data Management service. If you [turn off traffic for this endpoint](#), diagnostic and usage information, which helps Microsoft find and fix problems and improve our products and services, will not be sent back to Microsoft.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
svchost		cy2.vortex.data.microsoft.com.akadns.net	1709

The following endpoint is used by the Connected User Experiences and Telemetry component and connects to the Microsoft Data Management service. If you [turn off traffic for this endpoint](#), diagnostic and usage information, which helps Microsoft find and fix problems and improve our products and services, will not be sent back to Microsoft.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
svchost		v10.vortex-win.data.microsoft.com/collect/v1	1709

The following endpoints are used by Windows Error Reporting. To turn off traffic for these endpoints, enable the following Group Policy: Administrative Templates > Windows Components > Windows Error Reporting > Disable Windows Error Reporting. This means error reporting information will not be sent back to Microsoft.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
wermgr		watson.telemetry.microsoft.com	1709
	TLS v1.2	modern.watson.data.microsoft.com.akadns.net	1709

## Font streaming

The following endpoints are used to download fonts on demand. If you [turn off traffic for these endpoints](#), you will not be able to download fonts on demand.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
svchost		fs.microsoft.com	1709
		fs.microsoft.com/fs/windows/config.json	1709

## Licensing

The following endpoint is used for online activation and some app licensing. To turn off traffic for this endpoint, disable the Windows License Manager Service. This will also block online activation and app licensing may not work.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
licensemanager	HTTPS	licensing.mp.microsoft.com/v7.0/licenses/content	1709

## Location

The following endpoint is used for location data. If you [turn off traffic for this endpoint](#), apps cannot use location data.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
	HTTP	location-inference-westus.cloudapp.net	1709

## Maps

The following endpoint is used to check for updates to maps that have been downloaded for offline use. If you [turn off traffic for this endpoint](#), offline maps will not be updated.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
svchost	HTTPS	*g.akamaiedge.net	1709

## Microsoft account

The following endpoints are used for Microsoft accounts to sign in. If you [turn off traffic for these endpoints](#), users cannot sign in with Microsoft accounts.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
		login.msa.akadns6.net	1709
system32\Auth.Host.exe	HTTPS	auth.gfx.ms	1709

## Microsoft Store

The following endpoint is used for the Windows Push Notification Services (WNS). WNS enables third-party developers to send toast, tile, badge, and raw updates from their own cloud service. This provides a mechanism to deliver new updates to your users in a power-efficient and dependable way. If you [turn off traffic for this endpoint](#), push notifications will no longer work, including MDM device management, mail synchronization, settings synchronization.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
		*.wns.windows.com	1709

The following endpoint is used to revoke licenses for malicious apps in the Microsoft Store. To turn off traffic for this endpoint, either uninstall the app or [disable the Microsoft Store](#). If you disable the Microsoft store, other

Microsoft Store apps cannot be installed or updated. Additionally, the Microsoft Store won't be able to revoke malicious apps and users will still be able to open them.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
	HTTP	storecatalogrevocation.storequality.microsoft.com	1709

The following endpoints are used to download image files that are called when applications run (Microsoft Store or Inbox MSN Apps). If you [turn off traffic for these endpoints](#), the image files won't be downloaded, and apps cannot be installed or updated from the Microsoft Store. Additionally, the Microsoft Store won't be able to revoke malicious apps and users will still be able to open them.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
	HTTPS	img-prod-cms-rt-microsoft-com.akamaized.net	1709
backgroundtransferhost	HTTPS	store-images.microsoft.com	1803

The following endpoints are used to communicate with Microsoft Store. If you [turn off traffic for these endpoints](#), apps cannot be installed or updated from the Microsoft Store. Additionally, the Microsoft Store won't be able to revoke malicious apps and users will still be able to open them.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
	HTTP	storeedgefd.dsx.mp.microsoft.com	1709
	HTTP	pti.store.microsoft.com	1709
	TLS v1.2	cy2.*.md.mp.microsoft.com.*	1709
svchost	HTTPS	displaycatalog.mp.microsoft.com	1803

## Network Connection Status Indicator (NCSI)

Network Connection Status Indicator (NCSI) detects Internet connectivity and corporate network connectivity status. NCSI sends a DNS request and HTTP query to this endpoint to determine if the device can communicate with the Internet. If you [turn off traffic for this endpoint](#), NCSI won't be able to determine if the device is connected to the Internet and the network status tray icon will show a warning.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
	HTTP	<a href="http://www.msftconnecttest.com/connecttest.txt">www.msftconnecttest.com/connecttest.txt</a>	1709

The following endpoints are used to connect to the Office 365 portal's shared infrastructure, including Office Online. For more info, see [Office 365 URLs and IP address ranges](#). You can turn this off by removing all Microsoft Office apps and the Mail and Calendar apps. If you turn off traffic for these endpoints, users won't be able to save documents to the cloud or see their recently used documents.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
		*.a-msedge.net	1709
hxstr		*.c-msedge.net	1709
		*.e-msedge.net	1709
		*.s-msedge.net	1709
	HTTPS	ocos-office365-s2s.msedge.net	1803

The following endpoint is used to connect to the Office 365 portal's shared infrastructure, including Office Online. For more info, see [Office 365 URLs and IP address ranges](#). You can turn this off by removing all Microsoft Office apps and the Mail and Calendar apps. If you turn off traffic for these endpoints, users won't be able to save documents to the cloud or see their recently used documents.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
system32\Auth.Host.exe	HTTPS	outlook.office365.com	1709

The following endpoint is OfficeHub traffic used to get the metadata of Office apps. To turn off traffic for this endpoint, either uninstall the app or [disable the Microsoft Store](#). If you disable the Microsoft store, other Microsoft Store apps cannot be installed or updated. Additionally, the Microsoft Store won't be able to revoke malicious apps and users will still be able to open them.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
Windows Apps\Microsoft.Windows.Photos	HTTPS	client-office365-tas.msedge.net	1709

## OneDrive

The following endpoint is a redirection service that's used to automatically update URLs. If you [turn off traffic for this endpoint](#), anything that relies on g.live.com to get updated URL information will no longer work.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
onedrive	HTTP \ HTTPS	g.live.com/1rewlive5skydrive/ODSUProduction	1709

The following endpoint is used by OneDrive for Business to download and verify app updates. For more info, see [Office 365 URLs and IP address ranges](#). To turn off traffic for this endpoint, uninstall OneDrive for Business. In this case, your device will not be able to get OneDrive for Business app updates.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
onedrive	HTTPS	oneclient.sfx.ms	1709

## 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 may stop working.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
dmclient		cy2.settings.data.microsoft.com.akadns.net	1709

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 may stop working.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
dmclient	HTTPS	settings.data.microsoft.com	1709

The following endpoint is used as a way for apps to dynamically update their configuration. Apps such as Windows Connected User Experiences and Telemetry component and Windows Insider Program use it. If you [turn off traffic for this endpoint](#), an app that uses this endpoint may stop working.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
svchost	HTTPS	settings-win.data.microsoft.com	1709

## Skype

The following endpoint is used to retrieve Skype configuration values. To turn off traffic for this endpoint, either uninstall the app or [disable the Microsoft Store](#). If you disable the Microsoft store, other Microsoft Store apps cannot be installed or updated. Additionally, the Microsoft Store won't be able to revoke malicious apps and users will still be able to open them.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
microsoft.windowscommunicationsapps.exe	HTTPS	config.edge.skype.com	1709

## Windows Defender

The following endpoint is used for Windows Defender when Cloud-based Protection is enabled. If you [turn off traffic for this endpoint](#), the device will not use Cloud-based Protection.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
		wdcp.microsoft.com	1709

The following endpoints are used for Windows Defender definition updates. If you [turn off traffic for these endpoints](#), definitions will not be updated.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
		definitionupdates.microsoft.com	1709
MpCmdRun.exe	HTTPS	go.microsoft.com	1709

## Windows Spotlight

The following endpoints are used to retrieve Windows Spotlight metadata that describes content, such as references to image locations, as well as suggested apps, Microsoft account notifications, and Windows tips. If you [turn off traffic for these endpoints](#), Windows Spotlight will still try to deliver new lock screen images and updated content but it will fail; suggested apps, Microsoft account notifications, and Windows tips will not be downloaded. For more information, see [Windows Spotlight](#).

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
backgroundtaskhost	HTTPS	arc.msn.com	1709
backgroundtaskhost		g.msn.com.nsatc.net	1709
	TLS v1.2	*.search.msn.com	1709
	HTTPS	ris.api.iris.microsoft.com	1709
	HTTPS	query.prod.cms.rt.microsoft.com	1709

## Windows Update

The following endpoint is used for Windows Update downloads of apps and OS updates, including HTTP downloads or HTTP downloads blended with peers. If you [turn off traffic for this endpoint](#), Windows Update downloads will not be managed, as critical metadata that is used to make downloads more resilient is blocked. Downloads may be impacted by corruption (resulting in re-downloads of full files). Additionally, downloads of the same update by multiple devices on the same local network will not use peer devices for bandwidth reduction.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
svchost	HTTPS	*.prod.do.dsp.mp.microsoft.com	1709

The following endpoints are used to download operating system patches and updates. If you [turn off traffic for these endpoints](#), the device will not be able to download updates for the operating system.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
svchost	HTTP	*.windowsupdate.com	1709
	HTTP	fg.download.windowsupdate.com.c.footprint.net	1709

The following endpoint is used by the Highwinds Content Delivery Network to perform Windows updates. If you [turn off traffic for this endpoint](#), the device will not perform updates.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
		cds.d2s7q6s2.hwcdn.net	1709

The following endpoints are used by the Verizon Content Delivery Network to perform Windows updates. If you [turn off traffic for these endpoints](#), the device will not perform updates.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
	HTTP	*wac.phicdn.net	1709
		*wac.edgecastcdn.net	1709

The following endpoint is used to download apps and Windows Insider Preview builds from the Microsoft Store. Time Limited URL (TLU) is a mechanism for protecting the content. For example, it prevents someone from copying the URL and then getting access to the app that the person has not acquired). If you [turn off traffic for this endpoint](#), the updating functionality on this device is essentially in a disabled state, resulting in user unable to get apps from the Store, get latest version of Windows, and so on.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
svchost		*.tlu.dl.delivery.mp.microsoft.com.c.footprint.net	1709

The following endpoint is used to download apps from the Microsoft Store. It's used as part of calculating the right ranges for apps. If you [turn off traffic for this endpoint](#), users of the device will not able to get apps from the Microsoft Store.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
svchost		emdl.ws.microsoft.com	1709

The following endpoints enable connections to Windows Update, Microsoft Update, and the online services of the Store. If you [turn off traffic for these endpoints](#), the device will not be able to connect to Windows Update and Microsoft Update to help keep the device secure. Also, the device will not be able to acquire and update apps from the Store.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
svchost	HTTPS	fe2.update.microsoft.com	1709
svchost		fe3.delivery.mp.microsoft.com	1709
		fe3.delivery.dsp.mp.microsoft.com.nsatc.net	1709
svchost	HTTPS	sls.update.microsoft.com	1709
	HTTP	*.dl.delivery.mp.microsoft.com	1803

The following endpoint is used for content regulation. If you [turn off traffic for this endpoint](#), the Windows Update Agent will be unable to contact the endpoint and fallback behavior will be used. This may result in content being either incorrectly downloaded or not downloaded at all.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
svchost	HTTPS	tsfe.trafficshaping.dsp.mp.microsoft.com	1709

The following endpoints are used to download content. If you [turn off traffic for these endpoints](#), you will block any content from being downloaded.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
		a122.dscd.akamai.net	1709
		a1621.g.akamai.net	1709

## Microsoft forward link redirection service (FWLink)

The following endpoint is used by the Microsoft forward link redirection service (FWLink) to redirect permanent web links to their actual, sometimes transitory, URL. FWLinks are similar to URL shorteners, just longer.

If you disable this endpoint, Windows Defender won't be able to update its malware definitions; links from Windows and other Microsoft products to the Web won't work; and PowerShell updateable Help won't update. To disable the traffic, instead disable the traffic that's getting forwarded.

SOURCE PROCESS	PROTOCOL	DESTINATION	APPLIES FROM WINDOWS 10 VERSION
Various	HTTPS	go.microsoft.com	1709

## Other Windows 10 editions

To view endpoints for non-Enterprise Windows 10 editions, see:

- [Windows 10, version 1709, connection endpoints for non-Enterprise editions](#)



- [Windows 10, version 1803, connection endpoints for non-Enterprise editions](#)

## Related links

- [Office 365 URLs and IP address ranges](#)
- [Network infrastructure requirements for Microsoft Intune](#)

# Windows 10, version 1709, connection endpoints for non-Enterprise editions

6/29/2018 • 12 minutes to read • [Edit Online](#)

## Applies to

- Windows 10 Home, version 1709
- Windows 10 Professional, version 1709
- Windows 10 Education, version 1709

In addition to the endpoints listed for [Windows 10 Enterprise](#), the following endpoints are available on other editions of Windows 10, version 1709.

We used the following methodology to derive these network endpoints:

1. Set up the latest version of Windows 10 on a test virtual machine using the default settings.
2. Leave the devices running idle for a week (that is, a user is not interacting with the system/device).
3. Use globally accepted network protocol analyzer/capturing tools and log all background egress traffic.
4. Compile reports on traffic going to public IP addresses.
5. The test virtual machine was logged in using a local account and was not joined to a domain or Azure Active Directory.

### NOTE

Microsoft uses global load balancers that can appear in network trace-routes. For example, an endpoint for \*.akadns.net might be used to load balance requests to an Azure datacenter, which can change over time.

## Windows 10 Home

DESTINATION	PROTOCOL	DESCRIPTION
*.tlu.dl.delivery.mp.microsoft.com.c.footprint.net	HTTP	Enables connections to Windows Update.
*.wac.phicdn.net	HTTP	Used by the Verizon Content Delivery Network to perform Windows updates.
*.1.msftsrvcs.vo.llnwi.net	HTTP	Used for Windows Update downloads of apps and OS updates.
*.c-msedge.net	HTTP	Used by OfficeHub to get the metadata of Office apps.
*.delivery.dsp.mp.microsoft.com.nsatc.net	TLSv1.2	Enables connections to Windows Update.
*.dscd.akamai.net	HTTP	Used to download content.

<b>DESTINATION</b>	<b>PROTOCOL</b>	<b>DESCRIPTION</b>
*.dspg.akamaiedge.net	HTTP	Used to check for updates to maps that have been downloaded for offline use.
*.hwcdn.net	HTTP	Used by the Highwinds Content Delivery Network to perform Windows updates.
*.m1-msedge.net	TLSv1.2	Used by OfficeHub to get the metadata of Office apps.
*.search.msn.com	TLSv1.2	Used to retrieve Windows Spotlight metadata.
*.wac.edgecastcdn.net	TLSv1.2	Used by the Verizon Content Delivery Network to perform Windows updates.
*.wns.windows.com	TLSv1.2	Used for the Windows Push Notification Services (WNS).
*prod.do.dsp.mp.microsoft.com	TLSv1.2/HTTPS	Used for Windows Update downloads of apps and OS updates.
.g.akamaiedge.net	HTTP	Used to check for updates to maps that have been downloaded for offline use.
telecommand.telemetry.microsoft.com	HTTPS	Used by Windows Error Reporting.
2.dl.delivery.mp.microsoft.com	HTTP	Enables connections to Windows Update.
2.tlu.dl.delivery.mp.microsoft.com	HTTP	Enables connections to Windows Update.
arc.msn.com	HTTPS	Used to retrieve Windows Spotlight metadata.
arc.msn.com.nsatc.net	TLSv1.2	Used to retrieve Windows Spotlight metadata.
a-ring.msedge.net	HTTPS	Used by OfficeHub to get the metadata of Office apps.
au.download.windowsupdate.com	HTTP	Used to download operating system patches and updates.
b-ring.msedge.net	HTTPS	Used by OfficeHub to get the metadata of Office apps.
candycrushsoda.king.com	TLSv1.2	Used for Candy Crush Saga updates.
cdn.content.prod.cms.msn.com	HTTP	Used to retrieve Windows Spotlight metadata.

DESTINATION	PROTOCOL	DESCRIPTION
cdn.onenote.net	HTTP	Used for OneNote Live Tile.
client-office365-tas.msedge.net	HTTP	Used to connect to the Office 365 portal's shared infrastructure, including Office Online.
config.edge.skype.com	HTTP	Used to retrieve Skype configuration values.
ctldl.windowsupdate.com	HTTP	Used to download certificates that are publicly known to be fraudulent.
cy2.displaycatalog.md.mp.microsoft.com.akadns.net	TLSv1.2	Used to communicate with Microsoft Store.
cy2.licensing.md.mp.microsoft.com.akadns.net	TLSv1.2	Used to communicate with Microsoft Store.
cy2.purchase.md.mp.microsoft.com.akadns.net	TLSv1.2	Used to communicate with Microsoft Store.
cy2.settings.data.microsoft.com.akadns.net	TLSv1.2	Used as a way for apps to dynamically update their configuration.
cy2.vortex.data.microsoft.com.akadns.net	TLSv1.2	Used to retrieve Windows Insider Preview builds.
definitionupdates.microsoft.com	HTTPS	Used for Windows Defender definition updates.
displaycatalog.mp.microsoft.com	HTTPS	Used to communicate with Microsoft Store.
dl.delivery.mp.microsoft.com	HTTPS	Enables connections to Windows Update.
dual-a-0001.a.msedge.net	TLSv1.2	Used by OfficeHub to get the metadata of Office apps.
fe2.update.microsoft.com	HTTPS	Enables connections to Windows Update, Microsoft Update, and the online services of Microsoft Store.
fe2.update.microsoft.com.nsatc.net	TLSv1.2	Enables connections to Windows Update, Microsoft Update, and the online services of Microsoft Store.
fe3.delivery.dsp.mp.microsoft.com.nsatc.net	TLSv1.2/HTTPS	Enables connections to Windows Update, Microsoft Update, and the online services of Microsoft Store.
fg.download.windowsupdate.com.c.footprint.net	HTTP	Used to download operating system patches and updates.

DESTINATION	PROTOCOL	DESCRIPTION
fp.msedge.net	HTTPS	Used by OfficeHub to get the metadata of Office apps.
g.live.com/1rewlive5skydrive/	HTTPS	Used by a redirection service to automatically update URLs.
g.msn.com.nsatc.net	HTTP	Used to retrieve Windows Spotlight metadata.
geo-prod.do.dsp.mp.microsoft.com.nsatc.net	TLSv1.2	Enables connections to Windows Update.
go.microsoft.com	HTTPS	Used by a redirection service to automatically update URLs.
img-prod-cms-rt-microsoft-com.akamaized.net	HTTPS	Used to download image files that are called when applications run (Microsoft Store or Inbox MSN Apps).
*.login.msa.akadns6.net	TLSv1.2	Used for Microsoft accounts to sign in.
licensing.mp.microsoft.com	HTTPS	Used for online activation and some app licensing.
location-inference-westus.cloudapp.net	TLSv1.2	Used for location data.
login.live.com	HTTPS	Used to authenticate a device.
mediaredirect.microsoft.com	HTTPS	Used by the Groove Music app to update HTTP handler status.
modern.watson.data.microsoft.com.akadns.net	TLSv1.2	Used by Windows Error Reporting.
msftsrvcs.vo.llnwd.net	HTTP	Enables connections to Windows Update.
msnbot-*.search.msn.com	TLSv1.2	Used to retrieve Windows Spotlight metadata.
oem.twimg.com	HTTPS	Used for the Twitter Live Tile.
onedient.sfx.ms	HTTPS	Used by OneDrive for Business to download and verify app updates.
peer4-wst.msedge.net	HTTPS	Used by OfficeHub to get the metadata of Office apps.
pti.store.microsoft.com	HTTPS	Used to communicate with Microsoft Store.
pti.store.microsoft.com.unistore.akadns.net	TLSv1.2	Used to communicate with Microsoft Store.

DESTINATION	PROTOCOL	DESCRIPTION
purchase.mp.microsoft.com	HTTPS	Used to communicate with Microsoft Store.
ris.api.iris.microsoft.com.akadns.net	TLSv1.2/HTTPS	Used to retrieve Windows Spotlight metadata.
settings-win.data.microsoft.com	HTTPS	Used for Windows apps to dynamically update their configuration.
sls.update.microsoft.com.nsatc.net	TLSv1.2/HTTPS	Enables connections to Windows Update.
star-mini.c10r.facebook.com	TLSv1.2	Used for the Facebook Live Tile.
storecatalogrevocation.storequality.microsoft.com	HTTPS	Used to revoke licenses for malicious apps on the Microsoft Store.
storeedgefd.dsx.mp.microsoft.com	HTTPS	Used to communicate with Microsoft Store.
store-images.s-microsoft.com	HTTP	Used to get images that are used for Microsoft Store suggestions.
tile-service.weather.microsoft.com	HTTP	Used to download updates to the Weather app Live Tile.
tsfe.trafficshaping.dsp.mp.microsoft.com	TLSv1.2	Used for content regulation.
v10.vortex-win.data.microsoft.com	HTTPS	Used to retrieve Windows Insider Preview builds.
wallet.microsoft.com	HTTPS	Used by the Microsoft Wallet app.
wallet-frontend-prod-westus.cloudapp.net	TLSv1.2	Used by the Microsoft Wallet app.
watson.telemetry.microsoft.com	HTTPS	Used by Windows Error Reporting.
wdcp.microsoft.akadns.net	TLSv1.2	Used for Windows Defender when Cloud-based Protection is enabled.
wildcard.twimg.com	TLSv1.2	Used for the Twitter Live Tile.
<a href="http://www.bing.com">www.bing.com</a>	HTTP	Used for updates for Cortana, apps, and Live Tiles.
<a href="http://www.facebook.com">www.facebook.com</a>	HTTPS	Used for the Facebook Live Tile.
<a href="http://www.microsoft.com">www.microsoft.com</a>	HTTPS	Used for updates for Cortana, apps, and Live Tiles.

DESTINATION	PROTOCOL	DESCRIPTION
..akamai.net	HTTP	Used to download content.
..akamaiedge.net	TLSv1.2/HTTP	Used to check for updates to maps that have been downloaded for offline use.
*.a-msedge.net	TLSv1.2	Used by OfficeHub to get the metadata of Office apps.
*.blob.core.windows.net	HTTPS	Used by Windows Update to update words used for language input methods.
*.c-msedge.net	HTTP	Used by OfficeHub to get the metadata of Office apps.
*.dl.delivery.mp.microsoft.com	HTTP	Enables connections to Windows Update.
*.dspb.akamaiedge.net	TLSv1.2	Used to check for updates to maps that have been downloaded for offline use.
*.dspg.akamaiedge.net	TLSv1.2	Used to check for updates to maps that have been downloaded for offline use.
*.e-msedge.net	TLSv1.2	Used by OfficeHub to get the metadata of Office apps.
*.login.msa.akadns6.net	TLSv1.2	Used for Microsoft accounts to sign in.
*.s-msedge.net	TLSv1.2	Used by OfficeHub to get the metadata of Office apps.
*.telecommand.telemetry.microsoft.com.akadns.net	TLSv1.2	Used by Windows Error Reporting.
*.wac.edgecastcdn.net	TLSv1.2	Used by the Verizon Content Delivery Network to perform Windows updates.
*.wac.phicdn.net	HTTP	Used by the Verizon Content Delivery Network to perform Windows updates.
*.wns.windows.com	TLSv1.2	Used for the Windows Push Notification Services (WNS).
*prod.do.dsp.mp.microsoft.com	TLSv1.2/HTTPS	Used for Windows Update downloads of apps and OS updates.
3.dl.delivery.mp.microsoft.com	HTTPS	Enables connections to Windows Update.
3.dl.delivery.mp.microsoft.com.c.footprint.net	HTTP	Enables connections to Windows Update.

DESTINATION	PROTOCOL	DESCRIPTION
3.tlu.dl.delivery.mp.microsoft.com	HTTP	Enables connections to Windows Update.
3.tlu.dl.delivery.mp.microsoft.com.c.footprint.net	HTTP	Enables connections to Windows Update.
arc.msn.com	HTTPS	Used to retrieve Windows Spotlight metadata.
arc.msn.com.nsatc.net	TLSv1.3	Used to retrieve Windows Spotlight metadata.
au.download.windowsupdate.com	HTTPS	Used to download operating system patches and updates.
b-ring.msedge.net	HTTPS	Used by OfficeHub to get the metadata of Office apps.
candycrushsoda.king.com	HTTPS	Used for Candy Crush Saga updates.
cdn.content.prod.cms.msn.com	HTTP	Used to retrieve Windows Spotlight metadata.
cdn.onenote.net	HTTPS	Used for OneNote Live Tile.
client-office365-tas.msedge.net	HTTPS	Used to connect to the Office 365 portal's shared infrastructure, including Office Online.
config.edge.skype.com	HTTPS	Used to retrieve Skype configuration values.
ctldl.windowsupdate.com	HTTP	Used to download certificates that are publicly known to be fraudulent.
cs12.wpc.v0cdn.net	HTTP	Used by the Verizon Content Delivery Network to download content for Windows upgrades with Wireless Planning and Coordination (WPC).
cy2.displaycatalog.md.mp.microsoft.com.akadns.net	TLSv1.2	Used to communicate with Microsoft Store.
cy2.settings.data.microsoft.com.akadns.net	TLSv1.2	Used as a way for apps to dynamically update their configuration.
cy2.vortex.data.microsoft.com.akadns.net	TLSv1.2	Used to retrieve Windows Insider Preview builds.
definitionupdates.microsoft.com	HTTPS	Used for Windows Defender definition updates.
displaycatalog.mp.microsoft.com	HTTPS	Used to communicate with Microsoft Store.



DESTINATION	PROTOCOL	DESCRIPTION
download.windowsupdate.com	HTTP	Enables connections to Windows Update.
evoke-windowservices-tas.msedge.net	HTTPS	Used by the Photos app to download configuration files, and to connect to the Office 365 portal's shared infrastructure, including Office Online.
fe2.update.microsoft.com	HTTPS	Enables connections to Windows Update, Microsoft Update, and the online services of Microsoft Store.
fe2.update.microsoft.com.nsatc.net	TLSv1.2	Enables connections to Windows Update, Microsoft Update, and the online services of Microsoft Store.
fe3.delivery.dsp.mp.microsoft.com.nsatc.net	TLSv1.2/HTTPS	Enables connections to Windows Update, Microsoft Update, and the online services of Microsoft Store.
fe3.delivery.mp.microsoft.com	HTTPS	Enables connections to Windows Update, Microsoft Update, and the online services of Microsoft Store.
fg.download.windowsupdate.com.c.footprint.net	HTTP	Used to download operating system patches and updates.
fp.msedge.net	HTTPS	Used by OfficeHub to get the metadata of Office apps.
fs.microsoft.com	HTTPS	Used to download fonts on demand
g.live.com	HTTP	Used by a redirection service to automatically update URLs.
g.msn.com	HTTPS	Used to retrieve Windows Spotlight metadata.
g.msn.com.nsatc.net	TLSv1.2	Used to retrieve Windows Spotlight metadata.
geo-prod.do.dsp.mp.microsoft.com	HTTPS	Enables connections to Windows Update.
geover-prod.do.dsp.mp.microsoft.com	HTTPS	Enables connections to Windows Update.
go.microsoft.com	HTTPS	Used by a redirection service to automatically update URLs.
gpla1.wac.v2cdn.net	HTTP	Used for Baltimore CyberTrust Root traffic. .

<b>DESTINATION</b>	<b>PROTOCOL</b>	<b>DESCRIPTION</b>
img-prod-cms-rt-microsoft-com.akamaized.net	HTTPS	Used to download image files that are called when applications run (Microsoft Store or Inbox MSN Apps).
licensing.mp.microsoft.com	HTTPS	Used for online activation and some app licensing.
location-inference-westus.cloudapp.net	TLSv1.2	Used for location data.
login.live.com	HTTPS	Used to authenticate a device.
l-ring.msedge.net	HTTPS	Used by OfficeHub to get the metadata of Office apps.
mediaredirect.microsoft.com	HTTPS	Used by the Groove Music app to update HTTP handler status.
modern.watson.data.microsoft.com.aka dns.net	TLSv1.2	Used by Windows Error Reporting.
msnbot-*.search.msn.com	TLSv1.2	Used to retrieve Windows Spotlight metadata.
oem.twimg.com	HTTP	Used for the Twitter Live Tile.
onedient.sfx.ms	HTTP	Used by OneDrive for Business to download and verify app updates.
peer1-wst.msedge.net	HTTP	Used by OfficeHub to get the metadata of Office apps.
pti.store.microsoft.com	HTTPS	Used to communicate with Microsoft Store.
pti.store.microsoft.com.unistore.akadns.net	HTTPS	Used to communicate with Microsoft Store.
purchase.mp.microsoft.com	HTTPS	Used to communicate with Microsoft Store.
ris.api.iris.microsoft.com	HTTPS	Used to retrieve Windows Spotlight metadata.
settings-win.data.microsoft.com	HTTPS	Used for Windows apps to dynamically update their configuration.
sls.update.microsoft.com	HTTPS	Enables connections to Windows Update.
storecatalogrevocation.storequality.microsoft.com	HTTPS	Used to revoke licenses for malicious apps on the Microsoft Store.

DESTINATION	PROTOCOL	DESCRIPTION
storeedgefd.dsx.mp.microsoft.com	HTTPS	Used to communicate with Microsoft Store.
store-images.s-microsoft.com	HTTPS	Used to get images that are used for Microsoft Store suggestions.
store-images.s-microsoft.com	HTTPS	Used to get images that are used for Microsoft Store suggestions.
telecommand.telemetry.microsoft.com	HTTPS	Used by Windows Error Reporting.
tile-service.weather.microsoft.com	HTTP	Used to download updates to the Weather app Live Tile.
tsfe.trafficshaping.dsp.mp.microsoft.com	HTTPS	Used for content regulation.
v10.vortex-win.data.microsoft.com	HTTPS	Used to retrieve Windows Insider Preview builds.
wallet.microsoft.com	HTTPS	Used by the Microsoft Wallet app.
watson.telemetry.microsoft.com	HTTPS	Used by Windows Error Reporting.
wdcp.microsoft.akadns.net	HTTPS	Used for Windows Defender when Cloud-based Protection is enabled.
wildcard.twimg.com	TLSv1.2	Used for the Twitter Live Tile.
<a href="http://www.bing.com">www.bing.com</a>	TLSv1.2	Used for updates for Cortana, apps, and Live Tiles.
<a href="http://www.facebook.com">www.facebook.com</a>	HTTPS	Used for the Facebook Live Tile.
<a href="http://www.microsoft.com">www.microsoft.com</a>	HTTPS	Used for updates for Cortana, apps, and Live Tiles.

## Windows 10 Education

DESTINATION	PROTOCOL	DESCRIPTION
*.a-msedge.net	TLSv1.2	Used by OfficeHub to get the metadata of Office apps.
*.b.akamaiedge.net	TLSv1.2	Used to check for updates to maps that have been downloaded for offline use.
*.c-msedge.net	HTTP	Used by OfficeHub to get the metadata of Office apps.
*.dscb1.akamaiedge.net	HTTP	Used to check for updates to maps that have been downloaded for offline use.

DESTINATION	PROTOCOL	DESCRIPTION
*.dscd.akamai.net	HTTP	Used to download content.
*.dspb.akamaiedge.net	TLSv1.2	Used to check for updates to maps that have been downloaded for offline use.
*.dspw65.akamai.net	HTTP	Used to download content.
*.e-msedge.net	TLSv1.2	Used by OfficeHub to get the metadata of Office apps.
*.g.akamai.net	HTTP	Used to download content.
*.g.akamaiedge.net	TLSv1.2	Used to check for updates to maps that have been downloaded for offline use.
*.l.windowsupdate.com	HTTP	Enables connections to Windows Update.
*.s-msedge.net	TLSv1.2	Used by OfficeHub to get the metadata of Office apps.
*.wac.phicdn.net	HTTP	Used by the Verizon Content Delivery Network to perform Windows updates
*.wns.windows.com	TLSv1.2	Used for the Windows Push Notification Services (WNS).
*prod.do.dsp.mp.microsoft.com	TLSv1.2	Used for Windows Update downloads of apps and OS updates.
*prod.do.dsp.mp.microsoft.com.nsatc.net	TLSv1.2	Used for Windows Update downloads of apps and OS updates.
3.dl.delivery.mp.microsoft.com.c.footprint.net	HTTP	Enables connections to Windows Update.
3.tlu.dl.delivery.mp.microsoft.com.c.footprint.net	HTTP	Enables connections to Windows Update.
a-ring.msedge.net	HTTPS	Used by OfficeHub to get the metadata of Office apps.
au.download.windowsupdate.com	HTTP	Used to download operating system patches and updates.
cdn.onenote.net	HTTPS	Used for OneNote Live Tile.
cds.*.hwcdn.net	HTTP	Used by the Highwinds Content Delivery Network to perform Windows updates.
co4.telecommand.telemetry.microsoft.com.akadns.net	TLSv1.2	Used by Windows Error Reporting.

DESTINATION	PROTOCOL	DESCRIPTION
config.edge.skype.com	HTTPS	Used to retrieve Skype configuration values.
ctldl.windowsupdate.com	HTTP	Used to download certificates that are publicly known to be fraudulent.
cs12.wpc.v0cdn.net	HTTP	Used by the Verizon Content Delivery Network to download content for Windows upgrades with Wireless Planning and Coordination (WPC).
cy2.displaycatalog.md.mp.microsoft.com.akadns.net	TLSv1.2	Used to communicate with Microsoft Store.
cy2.settings.data.microsoft.com.akadns.net	TLSv1.2	Used as a way for apps to dynamically update their configuration.
cy2.vortex.data.microsoft.com.akadns.net	TLSv1.2	Used to retrieve Windows Insider Preview builds.
dl.delivery.mp.microsoft.com	HTTPS	Enables connections to Windows Update.
download.windowsupdate.com	HTTP	Enables connections to Windows Update.
evoke-windowservices-tas.msedge.net/ab	HTTPS	Used by the Photos app to download configuration files, and to connect to the Office 365 portal's shared infrastructure, including Office Online.
fe2.update.microsoft.com.nsatc.net	TLSv1.2	Enables connections to Windows Update, Microsoft Update, and the online services of Microsoft Store.
fe3.delivery.dsp.mp.microsoft.com.nsatc.net	TLSv1.2	Enables connections to Windows Update.
fg.download.windowsupdate.com.c.footprint.net	HTTP	Used to download operating system patches and updates.
fp.msedge.net	HTTPS	Used by OfficeHub to get the metadata of Office apps.
g.msn.com.nsatc.net	TLSv1.2/HTTP	Used to retrieve Windows Spotlight metadata.
geo-prod.do.dsp.mp.microsoft.com.nsatc.net	TLSv1.2	Enables connections to Windows Update.
geover-prod.do.dsp.mp.microsoft.com	HTTPS	Enables connections to Windows Update.

DESTINATION	PROTOCOL	DESCRIPTION
go.microsoft.com	HTTPS	Used by a redirection service to automatically update URLs.
gpla1.wac.v2cdn.net	HTTP	Used for Baltimore CyberTrust Root traffic. .
ipv4.login.msa.akadns6.net	TLSv1.2	Used for Microsoft accounts to sign in.
licensing.mp.microsoft.com	HTTPS	Used for online activation and some app licensing.
location-inference-westus.cloudapp.net	TLSv1.2	Used for location data.
login.live.com/*	HTTPS	Used to authenticate a device.
l-ring.msedge.net	HTTPS	Used by OfficeHub to get the metadata of Office apps.
mediaredirect.microsoft.com	HTTPS	Used by the Groove Music app to update HTTP handler status.
modern.watson.data.microsoft.com.aka dns.net	TLSv1.2	Used by Windows Error Reporting.
msftconnecttest.com/*	HTTP	Used by Network Connection Status Indicator (NCSI) to detect Internet connectivity and corporate network connectivity status.
msnbot-65-52-108-198.search.msn.com	TLSv1.2	Used to retrieve Windows Spotlight metadata.
onedient.sfx.ms	HTTP	Used by OneDrive for Business to download and verify app updates.
peer1-wst.msedge.net	HTTPS	Used by OfficeHub to get the metadata of Office apps.
pti.store.microsoft.com.unistore.akadns.net	TLSv1.2	Used to communicate with Microsoft Store.
settings-win.data.microsoft.com	HTTPS	Used for Windows apps to dynamically update their configuration.
sls.update.microsoft.com.nsatc.net	TLSv1.2	Enables connections to Windows Update.
store-images.s-microsoft.com	HTTPS	Used to get images that are used for Microsoft Store suggestions.
tile-service.weather.microsoft.com	HTTP	Used to download updates to the Weather app Live Tile.

<b>DESTINATION</b>	<b>PROTOCOL</b>	<b>DESCRIPTION</b>
telecommand.telemetry.microsoft.com	HTTPS	Used by Windows Error Reporting.
tsfe.trafficshaping.dsp.mp.microsoft.com	TLSv1.2	Used for content regulation.
wallet.microsoft.com	HTTPS	Used by the Microsoft Wallet app.
watson.telemetry.microsoft.com	HTTPS	Used by Windows Error Reporting.
wdcp.microsoft.akadns.net	TLSv1.2	Used for Windows Defender when Cloud-based Protection is enabled.
<a href="http://www.bing.com">www.bing.com</a>	HTTPS	Used for updates for Cortana, apps, and Live Tiles.

# Windows 10, version 1803, connection endpoints for non-Enterprise editions

7/2/2018 • 5 minutes to read • [Edit Online](#)

## Applies to

- Windows 10 Home, version 1803
- Windows 10 Professional, version 1803
- Windows 10 Education, version 1803

In addition to the endpoints listed for [Windows 10 Enterprise](#), the following endpoints are available on other editions of Windows 10, version 1803.

We used the following methodology to derive these network endpoints:

1. Set up the latest version of Windows 10 on a test virtual machine using the default settings.
2. Leave the devices running idle for a week (that is, a user is not interacting with the system/device).
3. Use globally accepted network protocol analyzer/capturing tools and log all background egress traffic.
4. Compile reports on traffic going to public IP addresses.
5. The test virtual machine was logged in using a local account and was not joined to a domain or Azure Active Directory.

### NOTE

Microsoft uses global load balancers that can appear in network trace-routes. For example, an endpoint for \*.akadns.net might be used to load balance requests to an Azure datacenter, which can change over time.

## Windows 10 Family

DESTINATION	PROTOCOL	DESCRIPTION
*.e-msedge.net	HTTPS	Used by OfficeHub to get the metadata of Office apps.
*.g.akamaiedge.net	HTTPS	Used to check for updates to maps that have been downloaded for offline use.
*.s-msedge.net	HTTPS	Used by OfficeHub to get the metadata of Office apps.
*.tlu.dl.delivery.mp.microsoft.com/filestreamingservice/files/	HTTP	Enables connections to Windows Update.
arc.msn.com.nsatc.net	HTTPS	Used to retrieve Windows Spotlight metadata.
arc.msn.com/v3/Delivery/Placement	HTTPS	Used to retrieve Windows Spotlight metadata.



DESTINATION	PROTOCOL	DESCRIPTION
client-office365-tas.msedge.net*	HTTPS	Used to connect to the Office 365 portal's shared infrastructure, including Office Online.
config.edge.skype.com/config/*	HTTPS	Used to retrieve Skype configuration values.
ctldl.windowsupdate.com/msdownload/update*	HTTP	Used to download certificates that are publicly known to be fraudulent.
cy2.displaycatalog.md.mp.microsoft.com.akadns.net	HTTPS	Used to communicate with Microsoft Store.
cy2.licensing.md.mp.microsoft.com.akadns.net	HTTPS	Used to communicate with Microsoft Store.
cy2.settings.data.microsoft.com.akadns.net	HTTPS	Used to communicate with Microsoft Store.
displaycatalog.mp.microsoft.com*	HTTPS	Used to communicate with Microsoft Store.
dm3p.wns.notify.windows.com.akadns.net	HTTPS	Used for the Windows Push Notification Services (WNS).
fe2.update.microsoft.com*	HTTPS	Enables connections to Windows Update, Microsoft Update, and the online services of Microsoft Store.
fe3.delivery.dsp.mp.microsoft.com.nsatc.net	HTTPS	Enables connections to Windows Update, Microsoft Update, and the online services of Microsoft Store.
fe3.delivery.mp.microsoft.com	HTTPS	Enables connections to Windows Update, Microsoft Update, and the online services of Microsoft Store.
g.live.com/odclientsettings/Prod	HTTPS	Used by OneDrive for Business to download and verify app updates.
g.msn.com.nsatc.net	HTTPS	Used to retrieve Windows Spotlight metadata.
geo-prod.dodsp.mp.microsoft.com.nsatc.net	HTTPS	Enables connections to Windows Update.
ipv4.login.msa.akadns6.net	HTTPS	Used for Microsoft accounts to sign in.
licensing.mp.microsoft.com/v7.0/licenses/content	HTTPS	Used for online activation and some app licensing.
location-inference-westus.cloudapp.net	HTTPS	Used for location data.

DESTINATION	PROTOCOL	DESCRIPTION
maps.windows.com/windows-app-web-link	HTTPS	Link to Maps application.
modern.watson.data.microsoft.com.aka dns.net	HTTPS	Used by Windows Error Reporting.
ocos-office365-s2s.msedge.net*	HTTPS	Used to connect to the Office 365 portal's shared infrastructure.
ocsp.digicert.com*	HTTP	CRL and OCSP checks to the issuing certificate authorities.
oneclient.sfx.ms*	HTTPS	Used by OneDrive for Business to download and verify app updates.
query.prod.cms.rt.microsoft.com*	HTTPS	Used to retrieve Windows Spotlight metadata.
ris.api.iris.microsoft.com*	HTTPS	Used to retrieve Windows Spotlight metadata.
settings.data.microsoft.com/settings/v2.0/*	HTTPS	Used for Windows apps to dynamically update their configuration.
settings-win.data.microsoft.com/settings/*	HTTPS	Used as a way for apps to dynamically update their configuration.
sls.update.microsoft.com*	HTTPS	Enables connections to Windows Update.
storecatalogrevocation.storequality.microsoft.com*	HTTPS	Used to revoke licenses for malicious apps on the Microsoft Store.
storeedgefd.dsx.mp.microsoft.com*	HTTPS	Used to communicate with Microsoft Store.
tile-service.weather.microsoft.com*	HTTP	Used to download updates to the Weather app Live Tile.
tsfe.trafficshaping.dsp.mp.microsoft.com	HTTPS	Used for content regulation.
ip5.afdorigin-prod-am02.afdogw.com	HTTPS	Used to serve office 365 experimentation traffic.
watson.telemetry.microsoft.com/Telemetry.Request	HTTPS	Used by Windows Error Reporting.

## Windows 10 Pro

DESTINATION	PROTOCOL	DESCRIPTION
-------------	----------	-------------

DESTINATION	PROTOCOL	DESCRIPTION
*.e-msedge.net	HTTPS	Used by OfficeHub to get the metadata of Office apps.
*.g.akamaiedge.net	HTTPS	Used to check for updates to maps that have been downloaded for offline use.
*.s-msedge.net	HTTPS	Used by OfficeHub to get the metadata of Office apps.
.tlu.dl.delivery.mp.microsoft.com/	HTTP	Enables connections to Windows Update.
*geo-prod.dodsp.mp.microsoft.com.nsatc.net	HTTPS	Enables connections to Windows Update.
arc.msn.com.nsatc.net	HTTPS	Used to retrieve Windows Spotlight metadata.
au.download.windowsupdate.com/*	HTTP	Enables connections to Windows Update.
ctldl.windowsupdate.com/msdownload/update/*	HTTP	Used to download certificates that are publicly known to be fraudulent.
cy2.licensing.md.mp.microsoft.com.akadns.net	HTTPS	Used to communicate with Microsoft Store.
cy2.settings.data.microsoft.com.akadns.net	HTTPS	Used to communicate with Microsoft Store.
dm3p.wns.notify.windows.com.akadns.net	HTTPS	Used for the Windows Push Notification Services (WNS)
fe3.delivery.dsp.mp.microsoft.com.nsatc.net	HTTPS	Enables connections to Windows Update, Microsoft Update, and the online services of Microsoft Store.
g.msn.com.nsatc.net	HTTPS	Used to retrieve Windows Spotlight metadata.
ipv4.login.msa.akadns6.net	HTTPS	Used for Microsoft accounts to sign in.
location-inference-westus.cloudapp.net	HTTPS	Used for location data.
modern.watson.data.microsoft.com.akadns.net	HTTPS	Used by Windows Error Reporting.
ocsp.digicert.com*	HTTP	CRL and OCSP checks to the issuing certificate authorities.
ris.api.iris.microsoft.com.akadns.net	HTTPS	Used to retrieve Windows Spotlight metadata.

DESTINATION	PROTOCOL	DESCRIPTION
tile-service.weather.microsoft.com/*	HTTP	Used to download updates to the Weather app Live Tile.
tsfe.trafficshaping.dsp.mp.microsoft.com	HTTPS	Used for content regulation.
vip5.afdorigin-prod-am02.afdogw.com	HTTPS	Used to serve office 365 experimentation traffic

## Windows 10 Education

DESTINATION	PROTOCOL	DESCRIPTION
*.b.akamaiedge.net	HTTPS	Used to check for updates to maps that have been downloaded for offline use.
*.e-msedge.net	HTTPS	Used by OfficeHub to get the metadata of Office apps.
*.g.akamaiedge.net	HTTPS	Used to check for updates to maps that have been downloaded for offline use.
*.s-msedge.net	HTTPS	Used by OfficeHub to get the metadata of Office apps.
*.telecommand.telemetry.microsoft.com. akadns.net	HTTPS	Used by Windows Error Reporting.
.tlu.dl.delivery.mp.microsoft.com	HTTP	Enables connections to Windows Update.
.windowsupdate.com	HTTP	Enables connections to Windows Update.
*geo-prod.do.dsp.mp.microsoft.com	HTTPS	Enables connections to Windows Update.
au.download.windowsupdate.com*	HTTP	Enables connections to Windows Update.
cdn.onenote.net/livetile/*	HTTPS	Used for OneNote Live Tile.
client-office365-tas.msedge.net/*	HTTPS	Used to connect to the Office 365 portal's shared infrastructure, including Office Online.
config.edge.skype.com/*	HTTPS	Used to retrieve Skype configuration values.
ctldl.windowsupdate.com/*	HTTP	Used to download certificates that are publicly known to be fraudulent.

DESTINATION	PROTOCOL	DESCRIPTION
cy2.displaycatalog.md.mp.microsoft.com.akadns.net	HTTPS	Used to communicate with Microsoft Store.
cy2.licensing.md.mp.microsoft.com.akadns.net	HTTPS	Used to communicate with Microsoft Store.
cy2.settings.data.microsoft.com.akadns.net	HTTPS	Used to communicate with Microsoft Store.
displaycatalog.mp.microsoft.com/*	HTTPS	Used to communicate with Microsoft Store.
download.windowsupdate.com/*	HTTPS	Enables connections to Windows Update.
emdl.ws.microsoft.com/*	HTTP	Used to download apps from the Microsoft Store.
fe2.update.microsoft.com/*	HTTPS	Enables connections to Windows Update, Microsoft Update, and the online services of Microsoft Store.
fe3.delivery.dsp.mp.microsoft.com.nsatc.net	HTTPS	Enables connections to Windows Update, Microsoft Update, and the online services of Microsoft Store.
fe3.delivery.mp.microsoft.com/*	HTTPS	Enables connections to Windows Update, Microsoft Update, and the online services of Microsoft Store.
g.live.com/odclientsettings/*	HTTPS	Used by OneDrive for Business to download and verify app updates.
g.msn.com.nsatc.net	HTTPS	Used to retrieve Windows Spotlight metadata.
ipv4.login.msa.akadns6.net	HTTPS	Used for Microsoft accounts to sign in.
licensing.mp.microsoft.com/*	HTTPS	Used for online activation and some app licensing.
maps.windows.com/windows-app-web-link	HTTPS	Link to Maps application
modern.watson.data.microsoft.com.akadns.net	HTTPS	Used by Windows Error Reporting.
ocos-office365-s2s.msedge.net/*	HTTPS	Used to connect to the Office 365 portal's shared infrastructure.
ocsp.digicert.com*	HTTP	CRL and OCSP checks to the issuing certificate authorities.

<b>DESTINATION</b>	<b>PROTOCOL</b>	<b>DESCRIPTION</b>
oneclient.sfx.ms/*	HTTPS	Used by OneDrive for Business to download and verify app updates.
settings-win.data.microsoft.com/settings/*	HTTPS	Used as a way for apps to dynamically update their configuration.
sls.update.microsoft.com/*	HTTPS	Enables connections to Windows Update.
storecatalogrevocation.storequality.microsoft.com/*	HTTPS	Used to revoke licenses for malicious apps on the Microsoft Store.
tile-service.weather.microsoft.com/*	HTTP	Used to download updates to the Weather app Live Tile.
tsfe.trafficshaping.dsp.mp.microsoft.com	HTTPS	Used for content regulation.
vip5.afdorigin-prod-ch02.afdogw.com	HTTPS	Used to serve office 365 experimentation traffic.
watson.telemetry.microsoft.com/Telemetry.Request	HTTPS	Used by Windows Error Reporting.
bing.com/*	HTTPS	Used for updates for Cortana, apps, and Live Tiles.

# Manage connections from Windows operating system components to Microsoft services

7/2/2018 • 59 minutes to read • [Edit Online](#)

## Applies to

- Windows 10 Enterprise, version 1607 and newer
- Windows Server 2016

If you're looking for content on what each diagnostic data level means and how to configure it in your organization, see [Configure Windows diagnostic data in your organization](#).

Learn about the network connections that Windows components make to Microsoft and also the privacy settings that affect data that is shared with either Microsoft or apps and how they can be managed by an IT Pro.

If you want to minimize connections from Windows to Microsoft services, or configure particular privacy settings, this article covers the settings that you could consider. You can configure diagnostic data at the lowest level for your edition of Windows, and also evaluate which other connections Windows makes to Microsoft services you want to turn off in your environment from the list in this article.

You can configure diagnostic data at the Security/Basic level, turn off Windows Defender diagnostic data and MSRT reporting, and turn off all other connections to Microsoft network endpoints as described in this article to help prevent Windows from sending any data to Microsoft. There are many reasons why these communications are enabled by default, such as updating malware definitions and maintain current certificate revocation lists, which is why we strongly recommend against this. This data helps us deliver a secure, reliable, and more delightful personalized experience.

To help make it easier to deploy settings to restrict connections from Windows 10 to Microsoft, you can apply the [Windows Restricted Traffic Limited Functionality Baseline](#). This baseline was created in the same way as the [Windows security baselines](#) that are often used to efficiently configure Windows to a known secure state. Running the Windows Restricted Traffic Limited Functionality Baseline on devices in your organization will allow you to quickly configure all of the settings covered in this document. However, some of the settings reduce the functionality and security configuration of your device and are therefore not recommended. Make sure should you've chosen the right settings configuration for your environment before applying. You should not extract this package to the windows\system32 folder because it will not apply correctly.

### IMPORTANT

As part of the [Windows Restricted Traffic Limited Functionality Baseline](#), MDM functionality is disabled. If you manage devices through MDM, make sure [cloud notifications are enabled](#).

Applying the Windows Restricted Traffic Limited Functionality Baseline is the same as applying each setting covered in this article. It is recommended that you restart a device after making configuration changes to it. Note that **Get Help** and **Give us Feedback** links no longer work after the Windows Restricted Traffic Limited Functionality Baseline is applied.

We are always striving to improve our documentation and welcome your feedback. You can provide feedback by contacting [telmhelp@microsoft.com](mailto:telmhelp@microsoft.com).

## What's new in Windows 10, version 1803 Enterprise edition

Here's a list of changes that were made to this article for Windows 10, version 1803:

- Added a policy to turn off notifications network usage
- Added a policy for Microsoft Edge to turn off configuration updates for the Books Library
- Added a policy for Microsoft Edge to turn off Address Bar drop-down list suggestions

## What's new in Windows 10, version 1709 Enterprise edition

Here's a list of changes that were made to this article for Windows 10, version 1709:

- Added the Phone calls section
- Added the Storage Health section
- Added discussion of apps for websites in the Microsoft Store section

## What's new in Windows 10, version 1703 Enterprise edition

Here's a list of changes that were made to this article for Windows 10, version 1703:

- Added an MDM policy for Font streaming
- Added an MDM policy for Network Connection Status Indicator
- Added an MDM policy for the Microsoft Account Sign-In Assistant
- Added instructions for removing the Sticky Notes app
- Added registry paths for some Group Policies
- Added the Find My Device section
- Added the Tasks section
- Added the App Diagnostics section
- Added the following Group Policies:
  - Prevent managing SmartScreen Filter
  - Turn off Compatibility View
  - Turn off Automatic Download and Install of updates
  - Do not connect to any Windows Update locations
  - Turn off access to all Windows Update features
  - Specify Intranet Microsoft update service location
  - Enable Windows NTP client
  - Turn off Automatic download of the ActiveX VersionList
  - Allow Automatic Update of Speech Data
  - Accounts: Block Microsoft Accounts
  - Do not use diagnostic data for tailored experiences

## Management options for each setting

The following sections list the components that make network connections to Microsoft services by default. You can configure these settings to control the data that is sent to Microsoft. To prevent Windows from sending any data to Microsoft, configure diagnostic data at the Security level, turn off Windows Defender diagnostic data and MSRT reporting, and turn off all of these connections.

### **Settings for Windows 10 Enterprise edition**



The following table lists management options for each setting, beginning with Windows 10 Enterprise version 1607.

**NOTE**

For some settings, MDM policies only partly cover capabilities available through Group Policy. See each setting's section for more details.

SETTING	UI	GROUP POLICY	MDM POLICY	REGISTRY	COMMAND LINE
1. Automatic Root Certificates Update		✓			
2. Cortana and Search	✓	✓	✓	✓	
3. Date & Time	✓	✓		✓	
4. Device metadata retrieval		✓		✓	
5. Find My Device		✓			
6. Font streaming		✓		✓	
7. Insider Preview builds	✓	✓	✓	✓	
8. Internet Explorer	✓	✓		✓	
9. Live Tiles		✓		✓	
10. Mail synchronization	✓		✓	✓	
11. Microsoft Account		✓	✓	✓	
12. Microsoft Edge	✓	✓	✓	✓	
13. Network Connection Status Indicator		✓		✓	
14. Offline maps	✓	✓		✓	

SETTING	UI	GROUP POLICY	MDM POLICY	REGISTRY	COMMAND LINE
15. OneDrive		✓		✓	
16. Preinstalled apps	✓				✓
17. Settings > Privacy					
17.1 General	✓	✓	✓	✓	
17.2 Location	✓	✓	✓	✓	
17.3 Camera	✓	✓	✓	✓	
17.4 Microphone	✓	✓	✓	✓	
17.5 Notifications	✓	✓	✓	✓	
17.6 Speech, inking, & typing	✓	✓	✓	✓	
17.7 Account info	✓	✓	✓	✓	
17.8 Contacts	✓	✓	✓	✓	
17.9 Calendar	✓	✓	✓	✓	
17.10 Call history	✓	✓	✓	✓	
17.11 Email	✓	✓	✓	✓	
17.12 Messaging	✓	✓	✓	✓	
17.13 Phone calls	✓	✓	✓	✓	
17.14 Radios	✓	✓	✓	✓	
17.15 Other devices	✓	✓	✓	✓	

SETTING	UI	GROUP POLICY	MDM POLICY	REGISTRY	COMMAND LINE
17.16 Feedback & diagnostics	✓	✓	✓	✓	
17.17 Background apps	✓	✓	✓		
17.18 Motion	✓	✓	✓	✓	
17.19 Tasks	✓	✓	✓	✓	
17.20 App Diagnostics	✓	✓	✓	✓	
18. Software Protection Platform		✓	✓	✓	
19. Storage Health		✓			
20. Sync your settings	✓	✓	✓	✓	
21. Tere do		✓		✓	✓
22. Wi-Fi Sense	✓	✓		✓	
23. Windows Defender		✓	✓	✓	
24. Windows Media Player	✓				✓
25. Windows Spotlight	✓	✓	✓	✓	
26. Microsoft Store		✓		✓	
26.1 Apps for websites		✓			
27. Windows Update Delivery Optimization	✓	✓	✓	✓	
28. Windows Update	✓	✓	✓		

See the following table for a summary of the management settings for Windows Server 2016 with Desktop Experience.

SETTING	UI	GROUP POLICY	REGISTRY	COMMAND LINE
1. Automatic Root Certificates Update		✓	✓	
2. Cortana and Search	✓	✓	✓	
3. Date & Time	✓	✓	✓	
4. Device metadata retrieval		✓	✓	
6. Font streaming		✓	✓	
7. Insider Preview builds	✓	✓	✓	
8. Internet Explorer	✓	✓	✓	
9. Live Tiles		✓	✓	
11. Microsoft Account		✓	✓	
13. Network Connection Status Indicator		✓	✓	
15. OneDrive		✓		
17. Settings > Privacy				
17.1 General	✓	✓	✓	
18. Software Protection Platform		✓	✓	
21. Teredo		✓	✓	✓
23. Windows Defender		✓	✓	
24. Windows Media Player				✓

SETTING	UI	GROUP POLICY	REGISTRY	COMMAND LINE
26. Microsoft Store		✓	✓	
26.1 Apps for websites		✓		
28. Windows Update		✓	✓	

### Settings for Windows Server 2016 Server Core

See the following table for a summary of the management settings for Windows Server 2016 Server Core.

SETTING	GROUP POLICY	REGISTRY	COMMAND LINE
1. Automatic Root Certificates Update	✓	✓	
3. Date & Time	✓	✓	
6. Font streaming	✓	✓	
13. Network Connection Status Indicator	✓		
18. Software Protection Platform	✓		
21. Teredo	✓		✓
23. Windows Defender	✓	✓	
28. Windows Update	✓	✓	

### Settings for Windows Server 2016 Nano Server

See the following table for a summary of the management settings for Windows Server 2016 Nano Server.

SETTING	REGISTRY	COMMAND LINE
1. Automatic Root Certificates Update	✓	
3. Date & Time	✓	
21. Teredo		✓
28. Windows Update	✓	

# How to configure each setting

Use the following sections for more information about how to configure each setting.

## 1. Automatic Root Certificates Update

The Automatic Root Certificates Update component is designed to automatically check the list of trusted authorities on Windows Update to see if an update is available. For more information, see [Automatic Root Certificates Update Configuration](#). Although not recommended, you can turn off Automatic Root Certificates Update, which also prevents updates to the disallowed certificate list and the pin rules list.

### Caution

By not automatically downloading the root certificates, the device might not be able to connect to some websites.

For Windows 10, Windows Server 2016 with Desktop Experience, and Windows Server 2016 Server Core:

- Enable the Group Policy: **Computer Configuration > Administrative Templates > System > Internet Communication Management > Internet Communication Settings > Turn off Automatic Root Certificates Update**

-and-

1. Navigate to **Computer Configuration > Windows Settings > Security Settings > Public Key Policies**.
2. Double-click **Certificate Path Validation Settings**.
3. On the **Network Retrieval** tab, select the **Define these policy settings** check box.
4. Clear the **Automatically update certificates in the Microsoft Root Certificate Program (recommended)** check box, and then click **OK**.

-or-

- Create the registry path **HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\SystemCertificates\AuthRoot** and then add a REG\_DWORD registry setting, named **DisableRootAutoUpdate**, with a value of 1.

-and-

1. Navigate to **Computer Configuration > Windows Settings > Security Settings > Public Key Policies**.
2. Double-click **Certificate Path Validation Settings**.
3. On the **Network Retrieval** tab, select the **Define these policy settings** check box.
4. Clear the **Automatically update certificates in the Microsoft Root Certificate Program (recommended)** check box, and then click **OK**.

On Windows Server 2016 Nano Server:

- Create the registry path **HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\SystemCertificates\AuthRoot** and then add a REG\_DWORD registry setting, named **DisableRootAutoUpdate**, with a value of 1.

### NOTE

CRL and OCSP network traffic is currently whitelisted and will still show up in network traces. CRL and OCSP checks are made to the issuing certificate authorities. Microsoft is one of them, but there are many others, such as DigiCert, Thawte, Google, Symantec, and VeriSign.

## 2. Cortana and Search

Use either Group Policy or MDM policies to manage settings for Cortana. For more info, see [Cortana, Search, and privacy: FAQ](#).

### 2.1 Cortana and Search Group Policies

Find the Cortana Group Policy objects under **Computer Configuration > Administrative Templates > Windows Components > Search**.

POLICY	DESCRIPTION
Allow Cortana	Choose whether to let Cortana install and run on the device.  Disable this policy to turn off Cortana.
Allow search and Cortana to use location	Choose whether Cortana and Search can provide location-aware search results.  Disable this policy to block access to location information for Cortana.
Do not allow web search	Choose whether to search the web from Windows Desktop Search.  Enable this policy to remove the option to search the Internet from Cortana.
Don't search the web or display web results in Search	Choose whether to search the web from Cortana.  Enable this policy to stop web queries and results from showing in Search.
Set what information is shared in Search	Control what information is shared with Bing in Search.  If you enable this policy and set it to <b>Anonymous info</b> , usage information will be shared but not search history, Microsoft Account information, or specific location.

You can also apply the Group Policies using the following registry keys:

POLICY	REGISTRY PATH
Allow Cortana	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Windows Search REG_DWORD: AllowCortana Value: 0
Allow search and Cortana to use location	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Windows Search REG_DWORD: AllowSearchToUseLocation Value: 0
Do not allow web search	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Windows Search REG_DWORD: DisableWebSearch Value: 1

POLICY	REGISTRY PATH
Don't search the web or display web results in Search	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Windows Search REG_DWORD: ConnectedSearchUseWeb Value: 0
Set what information is shared in Search	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Windows Search REG_DWORD: ConnectedSearchPrivacy Value: 3

In Windows 10, version 1507 and Windows 10, version 1511, when you enable the **Don't search the web or display web results in Search** Group Policy, you can control the behavior of whether Cortana searches the web to display web results. However, this policy only covers whether or not web search is performed. There could still be a small amount of network traffic to Bing.com to evaluate if certain Cortana components are up-to-date or not. In order to turn off that network activity completely, you can create a Windows Firewall rule to prevent outbound traffic.

#### IMPORTANT

These steps are not required for devices running Windows 10, version 1607 or Windows Server 2016.

1. Expand **Computer Configuration > Windows Settings > Security Settings > Windows Firewall with Advanced Security > Windows Firewall with Advanced Security - <LDAP name>**, and then click **Outbound Rules**.
2. Right-click **Outbound Rules**, and then click **New Rule**. The **New Outbound Rule Wizard** starts.
3. On the **Rule Type** page, click **Program**, and then click **Next**.
4. On the **Program** page, click **This program path**, type **%windir%\systemapps\Microsoft.Windows.Cortana\_cw5n1h2txyewy\SearchUI.exe**, and then click **Next**.
5. On the **Action** page, click **Block the connection**, and then click **Next**.
6. On the **Profile** page, ensure that the **Domain**, **Private**, and **Public** check boxes are selected, and then click **Next**.
7. On the **Name** page, type a name for the rule, such as **Cortana firewall configuration**, and then click **Finish**.
8. Right-click the new rule, click **Properties**, and then click **Protocols and Ports**.
9. Configure the **Protocols and Ports** page with the following info, and then click **OK**.
  - For **Protocol type**, choose **TCP**.
  - For **Local port**, choose **All Ports**.
  - For **Remote port**, choose **All ports**.

If your organization tests network traffic, do not use a network proxy as Windows Firewall does not block proxy traffic. Instead, use a network traffic analyzer. Based on your needs, there are many network traffic analyzers available at no cost.

## 2.2 Cortana and Search MDM policies



For Windows 10 only, the following Cortana MDM policies are available in the [Policy CSP](#).

POLICY	DESCRIPTION
Experience/AllowCortana	Choose whether to let Cortana install and run on the device.
Search/AllowSearchToUseLocation	Choose whether Cortana and Search can provide location-aware search results. Default: Allowed

### 3. Date & Time

You can prevent Windows from setting the time automatically.

- To turn off the feature in the UI: **Settings > Time & language > Date & time > Set time automatically**
- or-
- Create a REG\_SZ registry setting in **HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\Parameters\Type** with a value of **NoSync**.

After that, configure the following:

- Disable the Group Policy: **Computer Configuration > Administrative Templates > System > Enable Windows NTP Server > Windows Time Service > Configure Windows NTP Client**

#### NOTE

This is only available on Windows 10, version 1703 and later. If you're using Windows 10, version 1607, the Group Policy setting is **Computer Configuration > Administrative Templates > System > Windows Time Service > Time Providers > Enable Windows NTP Client**

-or -

- Create a new REG\_DWORD registry setting named **Enabled** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\W32time\TimeProviders\NtpClient** and set it to 0 (zero).

### 4. Device metadata retrieval

To prevent Windows from retrieving device metadata from the Internet, apply the Group Policy: **Computer Configuration > Administrative Templates > System > Device Installation > Prevent device metadata retrieval from the Internet**.

You can also create a new REG\_DWORD registry setting named **PreventDeviceMetadataFromNetwork** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Device Metadata** and set it to 1 (one).

### 5. Find My Device

To turn off Find My Device:

- Turn off the feature in the UI
- or-
- Disable the Group Policy: **Computer Configuration > Administrative Template > Windows Components > Find My Device > Turn On/Off Find My Device**

You can also create a new REG\_DWORD registry setting

**HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\FindMyDevice\AllowFindMyDevice** to 0 (zero).

## 6. Font streaming

Fonts that are included in Windows but that are not stored on the local device can be downloaded on demand.

If you're running Windows 10, version 1607, Windows Server 2016, or later:

- Disable the Group Policy: **Computer Configuration > Administrative Templates > Network > Fonts > Enable Font Providers**.
- Create a new REG\_DWORD registry setting **HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\Windows\System\EnableFontProviders** to 0 (zero).
- In Windows 10, version 1703, you can apply the System/AllowFontProviders MDM policy from the [Policy CSP](#) where:
  - **false**. Font streaming is disabled.
  - **true**. Font streaming is enabled.

If you're running Windows 10, version 1507 or Windows 10, version 1511, create a REG\_DWORD registry setting named **DisableFontProviders** in

**HKEY\_LOCAL\_MACHINE\System\CurrentControlSet\Services\FontCache\Parameters** with a value of 1.

### NOTE

After you apply this policy, you must restart the device for it to take effect.

## 7. Insider Preview builds

The Windows Insider Preview program lets you help shape the future of Windows, be part of the community, and get early access to releases of Windows 10. This setting stops communication with the Windows Insider Preview service that checks for new builds. Windows Insider Preview builds only apply to Windows 10 and are not available for Windows Server 2016.

### NOTE

If you upgrade a device that is configured to minimize connections from Windows to Microsoft services (that is, a device configured for zero exhaust) to a Windows Insider Preview build, the Feedback & Diagnostic setting will automatically be set to **Full**. Although the diagnostic data level may initially appear as **Basic**, a few hours after the UI is refreshed or the machine is rebooted, the setting will become **Full**.

To turn off Insider Preview builds for a released version of Windows 10:

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Data Collection and Preview Builds > Toggle user control over Insider builds**.

To turn off Insider Preview builds for Windows 10:

### NOTE

If you're running a preview version of Windows 10, you must roll back to a released version before you can turn off Insider Preview builds.

- Turn off the feature in the UI: **Settings > Update & security > Windows Insider Program > Stop Insider Preview builds.**

-or-

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Data Collection and Preview Builds > Toggle user control over Insider builds.**

-or -

- Create a new REG\_DWORD registry setting named **AllowBuildPreview** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\Windows\PreviewBuilds** with a value of 0 (zero)

-or-

- Apply the System/AllowBuildPreview MDM policy from the [Policy CSP](#) where:
  - **0.** Users cannot make their devices available for downloading and installing preview software.
  - **1.** Users can make their devices available for downloading and installing preview software.
  - **2.** (default) Not configured. Users can make their devices available for download and installing preview software.

-or-

- Create a provisioning package: **Runtime settings > Policies > System > AllowBuildPreview**, where:
  - **0.** Users cannot make their devices available for downloading and installing preview software.
  - **1.** Users can make their devices available for downloading and installing preview software.
  - **2.** (default) Not configured. Users can make their devices available for download and installing preview software.

## 8. Internet Explorer

Use Group Policy to manage settings for Internet Explorer. You can find the Internet Explorer Group Policy objects under **Computer Configuration > Administrative Templates > Windows Components > Internet Explorer.**

POLICY	DESCRIPTION
Turn on Suggested Sites	Choose whether an employee can configure Suggested Sites. Default: Enabled You can also turn this off in the UI by clearing the <b>Internet Options &gt; Advanced &gt; Enable Suggested Sites</b> check box.
Allow Microsoft services to provide enhanced suggestions as the user types in the Address Bar	Choose whether an employee can configure enhanced suggestions, which are presented to the employee as they type in the Address Bar. Default: Enabled
Turn off the auto-complete feature for web addresses	Choose whether auto-complete suggests possible matches when employees are typing web address in the Address Bar. Default: Disabled You can also turn this off in the UI by clearing the <b>Internet Options &gt; Advanced &gt; Use inline AutoComplete in the Internet Explorer Address Bar and Open Dialog</b> check box.

POLICY	DESCRIPTION
Turn off browser geolocation	Choose whether websites can request location data from Internet Explorer. Default: Disabled
Prevent managing SmartScreen filter	Choose whether employees can manage the SmartScreen Filter in Internet Explorer. Default: Disabled

Alternatively, you could use the registry to set the Group Policies.

POLICY	REGISTRY PATH
Turn on Suggested Sites	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Internet Explorer\Suggested Sites REG_DWORD: Enabled Value: 0
Allow Microsoft services to provide enhanced suggestions as the user types in the Address Bar	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Internet Explorer REG_DWORD: AllowServicePoweredQSA Value: 0
Turn off the auto-complete feature for web addresses	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Internet Explorer\AutoComplete REG_SZ: AutoSuggest Value: <b>No</b>
Turn off browser geolocation	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Internet Explorer\Geolocation REG_DWORD: PolicyDisableGeolocation Value: 1
Prevent managing SmartScreen filter	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Internet Explorer\PhishingFilter REG_DWORD: EnabledV9 Value: 0

There are three more Group Policy objects that are used by Internet Explorer:

PATH	POLICY	DESCRIPTION
<b>Computer Configuration &gt; Administrative Templates &gt; Windows Components &gt; Internet Explorer &gt; Compatibility View &gt; Turn off Compatibility View</b>	Choose whether employees can configure Compatibility View.	Choose whether an employee can swipe across a screen or click forward to go to the next pre-loaded page of a website. Default: Disabled
<b>Computer Configuration &gt; Administrative Templates &gt; Windows Components &gt; Internet Explorer &gt; Internet Control Panel &gt; Advanced Page</b>	Turn off the flip ahead with page prediction feature	Choose whether an employee can swipe across a screen or click forward to go to the next pre-loaded page of a website. Default: Enabled

PATH	POLICY	DESCRIPTION
<b>Computer Configuration</b> > <b>Administrative Templates</b> > <b>Windows Components</b> > <b>RSS Feeds</b>	Turn off background synchronization for feeds and Web Slices	Choose whether to have background synchronization for feeds and Web Slices. Default: Enabled

You can also use registry entries to set these Group Policies.

POLICY	REGISTRY PATH
Choose whether employees can configure Compatibility View.	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\MicrosoftEdge\BrowserEmulation REG_DWORD: MSCompatibilityMode Value: 0
Turn off the flip ahead with page prediction feature	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Internet Explorer\FlipeAhead REG_DWORD: Enabled Value: 0
Turn off background synchronization for feeds and Web Slices	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Internet Explorer\Feeds REG_DWORD: BackgroundSyncStatus Value: 0

To turn off the home page, enable the Group Policy: **User Configuration** > **Administrative Templates** > **Windows Components** > **Internet Explorer** > **Disable changing home page settings**, and set it to **about:blank**.

To configure the First Run Wizard, enable the Group Policy: **User Configuration** > **Administrative Templates** > **Windows Components** > **Internet Explorer** > **Prevent running First Run wizard**, and set it to **Go directly to home page**.

To configure the behavior for a new tab, enable the Group Policy: **User Configuration** > **Administrative Templates** > **Windows Components** > **Internet Explorer** > **Specify default behavior for a new tab**, and set it to **about:blank**.

### 8.1 ActiveX control blocking

ActiveX control blocking periodically downloads a new list of out-of-date ActiveX controls that should be blocked.

You can turn this off by:

- Apply the Group Policy: **User Configuration** > **Administrative Templates** > **Windows Components** > **Internet Explorer** > **Security Features** > **Add-on Management** > **Turn off Automatic download of the ActiveX VersionList**

-or -

- Changing the REG\_DWORD registry setting **HKEY\_CURRENT\_USER\Software\Microsoft\Internet Explorer\VersionManager\DownloadVersionList** to 0 (zero).

For more info, see [Out-of-date ActiveX control blocking](#).

### 9. Live Tiles

To turn off Live Tiles:

- Apply the Group Policy: **User Configuration** > **Administrative Templates** > **Start Menu and Taskbar**

> **Notifications** > **Turn Off notifications network usage**

-or-

- Create a REG\_DWORD registry setting named **NoCloudApplicationNotification** in **HKEY\_CURRENT\_USER\SOFTWARE\Policies\Microsoft\Windows\CurrentVersion\PushNotifications** with a value of 1 (one).

In Windows 10 Mobile, you must also unpin all tiles that are pinned to Start.

## 10. Mail synchronization

To turn off mail synchronization for Microsoft Accounts that are configured on a device:

- In **Settings** > **Accounts** > **Your email and accounts**, remove any connected Microsoft Accounts.
- or-
- Remove any Microsoft Accounts from the Mail app.
- or-
- Apply the Accounts/AllowMicrosoftAccountConnection MDM policy from the [Policy CSP](#) where 0 is not allowed and 1 is allowed. This does not apply to Microsoft Accounts that have already been configured on the device.

To turn off the Windows Mail app:

- Apply the Group Policy: **Computer Configuration** > **Administrative Templates** > **Windows Components** > **Windows Mail** > **Turn off Windows Mail application**
- or-
- Create a REG\_DWORD registry setting named **ManualLaunchAllowed** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\Windows Mail** with a value of 0 (zero).

## 11. Microsoft Account

To prevent communication to the Microsoft Account cloud authentication service. Many apps and system components that depend on Microsoft Account authentication may lose functionality. Some of them could be in unexpected ways.

- Apply the Group Policy: **Computer Configuration** > **Windows Settings** > **Security Settings** > **Local Policies** > **Security Options** > **Accounts: Block Microsoft Accounts** and set it to **Users can't add Microsoft accounts**.
- or-
- Create a REG\_DWORD registry setting named **NoConnectedUser** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System** with a value of 3. To disable the Microsoft Account Sign-In Assistant:
  - Apply the Accounts/AllowMicrosoftAccountSignInAssistant MDM policy from the [Policy CSP](#) where 0 is turned off and 1 is turned on.
  - Change the Start REG\_DWORD registry setting in **HKEY\_LOCAL\_MACHINE\System\CurrentControlSet\Services\wlidsvc** to a value of 4.

## 12. Microsoft Edge

Use either Group Policy or MDM policies to manage settings for Microsoft Edge. For more info, see [Microsoft Edge and privacy: FAQ](#).

## 12.1 Microsoft Edge Group Policies

Find the Microsoft Edge Group Policy objects under **Computer Configuration > Administrative Templates > Windows Components > Microsoft Edge**.

POLICY	DESCRIPTION
Allow configuration updates for the Books Library	Choose whether configuration updates are done for the Books Library. Default: Enabled
Configure Autofill	Choose whether employees can use autofill on websites. Default: Enabled
Configure Do Not Track	Choose whether employees can send Do Not Track headers. Default: Disabled
Configure Password Manager	Choose whether employees can save passwords locally on their devices. Default: Enabled
Configure search suggestions in Address Bar	Choose whether the Address Bar shows search suggestions. Default: Enabled
Configure Windows Defender SmartScreen Filter (Windows 10, version 1703) Configure SmartScreen Filter (Windows Server 2016)	Choose whether Windows Defender SmartScreen is turned on or off. Default: Enabled
Allow web content on New Tab page	Choose whether a new tab page appears. Default: Enabled
Configure Start pages	Choose the Start page for domain-joined devices. Set this to <b>&lt;about:blank&gt;</b>
Prevent the First Run webpage from opening on Microsoft Edge	Choose whether employees see the First Run webpage. Default: Disabled

The Windows 10, version 1511 Microsoft Edge Group Policy names are:

POLICY	DESCRIPTION
Allow address bar drop-down list suggestions	Choose whether employees can use Address Bar drop-down list suggestions. Default: Disabled
Turn off autofill	Choose whether employees can use autofill on websites. Default: Enabled
Allow employees to send Do Not Track headers	Choose whether employees can send Do Not Track headers. Default: Disabled
Turn off password manager	Choose whether employees can save passwords locally on their devices. Default: Enabled
Turn off Address Bar search suggestions	Choose whether the Address Bar shows search suggestions. Default: Enabled

POLICY	DESCRIPTION
Turn off the SmartScreen Filter	Choose whether SmartScreen is turned on or off. Default: Enabled
Open a new tab with an empty tab	Choose whether a new tab page appears. Default: Enabled
Configure corporate Home pages	Choose the corporate Home page for domain-joined devices. Set this to <b>about:blank</b>

Alternatively, you can configure the Microsoft Group Policies using the following registry entries:

POLICY	REGISTRY PATH
Allow Address Bar drop-down list suggestions	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\MicrosoftEdge\ServiceUI REG_DWORD name: ShowOneBox Value: 0
Allow configuration updates for the Books Library	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\MicrosoftEdge\BooksLibrary REG_DWORD name: AllowConfigurationUpdateForBooksLibrary Value: 1
Configure Autofill	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\MicrosoftEdge\Main REG_SZ name: UseFormSuggest Value : <b>no</b>
Configure Do Not Track	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\MicrosoftEdge\Main REG_DWORD name: DoNotTrack REG_DWORD: 1
Configure Password Manager	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\MicrosoftEdge\Main REG_SZ name: FormSuggest Passwords REG_SZ: <b>no</b>
Configure search suggestions in Address Bar	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\MicrosoftEdge\SearchScopes REG_DWORD name: ShowSearchSuggestionsGlobal Value: 0
Configure Windows Defender SmartScreen Filter (Windows 10, version 1703)	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\MicrosoftEdge\PhishingFilter REG_DWORD name: EnabledV9 Value: 0
Allow web content on New Tab page	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\MicrosoftEdge\SearchScopes REG_DWORD name: AllowWebContentOnNewTabPage Value: 0



POLICY	REGISTRY PATH
Configure corporate Home pages	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\MicrosoftEdge\ServiceUI REG_DWORD name: ProvisionedHomePages Value: 0

## 12.2 Microsoft Edge MDM policies

The following Microsoft Edge MDM policies are available in the [Policy CSP](#).

POLICY	DESCRIPTION
Browser/AllowAutoFill	Choose whether employees can use autofill on websites. Default: Allowed
Browser/AllowDoNotTrack	Choose whether employees can send Do Not Track headers. Default: Not allowed
Browser/AllowMicrosoftCompatibilityList	Specify the Microsoft compatibility list in Microsoft Edge. Default: Enabled
Browser/AllowPasswordManager	Choose whether employees can save passwords locally on their devices. Default: Allowed
Browser/AllowSearchSuggestionsinAddressBar	Choose whether the Address Bar shows search suggestions.. Default: Allowed
Browser/AllowSmartScreen	Choose whether SmartScreen is turned on or off. Default: Allowed
Browser/FirstRunURL	Choose the home page for Microsoft Edge on Windows Mobile 10. Default: blank

For a complete list of the Microsoft Edge policies, see [Available policies for Microsoft Edge](#).

## 13. Network Connection Status Indicator

Network Connection Status Indicator (NCSI) detects Internet connectivity and corporate network connectivity status. NCSI sends a DNS request and HTTP query to <http://www.msftconnecttest.com/connecttest.txt> to determine if the device can communicate with the Internet. For more info about NCSI, see [The Network Connection Status Icon](#).

In versions of Windows 10 prior to Windows 10, version 1607 and Windows Server 2016, the URL was .

You can turn off NCSI by doing one of the following:

- Enable the Group Policy: **Computer Configuration > Administrative Templates > System > Internet Communication Management > Internet Communication Settings > Turn off Windows Network Connectivity Status Indicator active tests**
- In Windows 10, version 1703 and later, apply the Connectivity/DisallowNetworkConnectivityActiveTests MDM policy.

#### NOTE

After you apply this policy, you must restart the device for the policy setting to take effect.

-or-

- Create a REG\_DWORD registry setting named **NoActiveProbe** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\Windows\NetworkConnectivityStatusIndicator** with a value of 1 (one).

#### 14. Offline maps

You can turn off the ability to download and update offline maps.

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Maps > Turn off Automatic Download and Update of Map Data**

-or-

- Create a REG\_DWORD registry setting named **AutoDownloadAndUpdateMapData** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Maps** with a value of 0 (zero).

-and-

- In Windows 10, version 1607 and later, apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Maps > Turn off unsolicited network traffic on the Offline Maps settings page**

-or-

- Create a REG\_DWORD registry setting named **AllowUntriggeredNetworkTrafficOnSettingsPage** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Maps** with a value of 0 (zero).

#### 15. OneDrive

To turn off OneDrive in your organization:

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > OneDrive > Prevent the usage of OneDrive for file storage**

-or-

- Create a REG\_DWORD registry setting named **DisableFileSyncNGSC** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\Windows\OneDrive** with a value of 1 (one).

-and-

- Create a REG\_DWORD registry setting named **PreventNetworkTrafficPreUserSignIn** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\OneDrive** with a value of 1 (one).

#### 16. Preinstalled apps

Some preinstalled apps get content before they are opened to ensure a great experience. You can remove these using the steps in this section.

To remove the News app:

- Right-click the app in Start, and then click **Uninstall**.

-or-

- Remove the app for new user accounts. From an elevated command prompt, run the following Windows

PowerShell command: **Get-AppxProvisionedPackage -Online | Where-Object {\$\_.PackageName - Like "Microsoft.BingNews"} | ForEach-Object { Remove-AppxProvisionedPackage -Online - PackageName \$\_.PackageName}**

-and-

Remove the app for the current user. From an elevated command prompt, run the following Windows

PowerShell command: **Get-AppxPackage Microsoft.BingNews | Remove-AppxPackage**

To remove the Weather app:

- Remove the app for new user accounts. From an elevated command prompt, run the following Windows PowerShell command: **Get-AppxProvisionedPackage -Online | Where-Object {\$\_.PackageName - Like "Microsoft.BingWeather"} | ForEach-Object { Remove-AppxProvisionedPackage -Online - PackageName \$\_.PackageName}**

-and-

Remove the app for the current user. From an elevated command prompt, run the following Windows

PowerShell command: **Get-AppxPackage Microsoft.BingWeather | Remove-AppxPackage**

To remove the Money app:

- Right-click the app in Start, and then click **Uninstall**.

-or-

- Remove the app for new user accounts. From an elevated command prompt, run the following Windows PowerShell command: **Get-AppxProvisionedPackage -Online | Where-Object {\$\_.PackageName - Like "Microsoft.BingFinance"} | ForEach-Object { Remove-AppxProvisionedPackage -Online - PackageName \$\_.PackageName}**

-and-

Remove the app for the current user. From an elevated command prompt, run the following Windows

PowerShell command: **Get-AppxPackage Microsoft.BingFinance | Remove-AppxPackage**

To remove the Sports app:

- Right-click the app in Start, and then click **Uninstall**.

-or-

- Remove the app for new user accounts. From an elevated command prompt, run the following Windows PowerShell command: **Get-AppxProvisionedPackage -Online | Where-Object {\$\_.PackageName - Like "Microsoft.BingSports"} | ForEach-Object { Remove-AppxProvisionedPackage -Online - PackageName \$\_.PackageName}**

-and-

Remove the app for the current user. From an elevated command prompt, run the following Windows

PowerShell command: **Get-AppxPackage Microsoft.BingSports | Remove-AppxPackage**

To remove the Twitter app:

- Right-click the app in Start, and then click **Uninstall**.

-or-

- Remove the app for new user accounts. From an elevated command prompt, run the following Windows PowerShell command: **Get-AppxProvisionedPackage -Online | Where-Object {\$\_.PackageName -**

**Like "\*.Twitter"} | ForEach-Object { Remove-AppxProvisionedPackage -Online -PackageName \$\_.PackageName}**

-and-

Remove the app for the current user. From an elevated command prompt, run the following Windows PowerShell command: **Get-AppxPackage \*.Twitter | Remove-AppxPackage**

To remove the XBOX app:

- Remove the app for new user accounts. From an elevated command prompt, run the following Windows PowerShell command: **Get-AppxProvisionedPackage -Online | Where-Object {\$\_.PackageName -Like "Microsoft.XboxApp"} | ForEach-Object { Remove-AppxProvisionedPackage -Online -PackageName \$\_.PackageName}**

-and-

Remove the app for the current user. From an elevated command prompt, run the following Windows PowerShell command: **Get-AppxPackage Microsoft.XboxApp | Remove-AppxPackage**

To remove the Sway app:

- Right-click the app in Start, and then click **Uninstall**.

-or-

- Remove the app for new user accounts. From an elevated command prompt, run the following Windows PowerShell command: **Get-AppxProvisionedPackage -Online | Where-Object {\$\_.PackageName -Like "Microsoft.Office.Sway"} | ForEach-Object { Remove-AppxProvisionedPackage -Online -PackageName \$\_.PackageName}**

-and-

Remove the app for the current user. From an elevated command prompt, run the following Windows PowerShell command: **Get-AppxPackage Microsoft.Office.Sway | Remove-AppxPackage**

To remove the OneNote app:

- Remove the app for new user accounts. From an elevated command prompt, run the following Windows PowerShell command: **Get-AppxProvisionedPackage -Online | Where-Object {\$\_.PackageName -Like "Microsoft.Office.OneNote"} | ForEach-Object { Remove-AppxProvisionedPackage -Online -PackageName \$\_.PackageName}**

-and-

Remove the app for the current user. From an elevated command prompt, run the following Windows PowerShell command: **Get-AppxPackage Microsoft.Office.OneNote | Remove-AppxPackage**

To remove the Get Office app:

- Right-click the app in Start, and then click **Uninstall**.

-or-

- Remove the app for new user accounts. From an elevated command prompt, run the following Windows PowerShell command: **Get-AppxProvisionedPackage -Online | Where-Object {\$\_.PackageName -Like "Microsoft.MicrosoftOfficeHub"} | ForEach-Object { Remove-AppxProvisionedPackage -Online -PackageName \$\_.PackageName}**

-and-

Remove the app for the current user. From an elevated command prompt, run the following Windows PowerShell command: **Get-AppxPackage Microsoft.MicrosoftOfficeHub | Remove-AppxPackage**

To remove the Get Skype app:

- Right-click the Sports app in Start, and then click **Uninstall**.

-or-

- Remove the app for new user accounts. From an elevated command prompt, run the following Windows PowerShell command: **Get-AppxProvisionedPackage -Online | Where-Object {\$\_.PackageName - Like "Microsoft.SkypeApp"} | ForEach-Object { Remove-AppxProvisionedPackage -Online -PackageName \$\_.PackageName }**

-and-

Remove the app for the current user. From an elevated command prompt, run the following Windows PowerShell command: **Get-AppxPackage Microsoft.SkypeApp | Remove-AppxPackage**

To remove the Sticky notes app:

- Remove the app for new user accounts. From an elevated command prompt, run the following Windows PowerShell command: **Get-AppxProvisionedPackage -Online | Where-Object {\$\_.PackageName - Like "Microsoft.MicrosoftStickyNotes"} | ForEach-Object { Remove-AppxProvisionedPackage -Online -PackageName \$\_.PackageName }**

-and-

Remove the app for the current user. From an elevated command prompt, run the following Windows PowerShell command: **Get-AppxPackage Microsoft.MicrosoftStickyNotes | Remove-AppxPackage**

## 17. Settings > Privacy

Use Settings > Privacy to configure some settings that may be important to your organization. Except for the Feedback & Diagnostics page, these settings must be configured for every user account that signs into the PC.

- [17.1 General](#)
- [17.2 Location](#)
- [17.3 Camera](#)
- [17.4 Microphone](#)
- [17.5 Notifications](#)
- [17.6 Speech, inking, & typing](#)
- [17.7 Account info](#)
- [17.8 Contacts](#)
- [17.9 Calendar](#)
- [17.10 Call history](#)
- [17.11 Email](#)
- [17.12 Messaging](#)
- [17.13 Radios](#)
- [17.14 Other devices](#)

- [17.15 Feedback & diagnostics](#)
- [17.16 Background apps](#)
- [17.17 Motion](#)
- [17.18 Tasks](#)
- [17.19 App Diagnostics](#)

## 17.1 General

**General** includes options that don't fall into other areas.

### Windows 10, version 1703 options

To turn off **Let apps use advertising ID to make ads more interesting to you based on your app usage (turning this off will reset your ID)**:

#### NOTE

When you turn this feature off in the UI, it turns off the advertising ID, not just resets it.

- Turn off the feature in the UI.
- or-
- Apply the Group Policy: **Computer Configuration > Administrative Templates > System > User Profiles > Turn off the advertising ID.**
- or-
- Create a REG\_DWORD registry setting named **Enabled** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\AdvertisingInfo** with a value of 0 (zero).
- or-
- Create a REG\_DWORD registry setting named **DisabledByGroupPolicy** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\Windows\AdvertisingInfo** with a value of 1 (one).

To turn off **Let websites provide locally relevant content by accessing my language list**:

- Turn off the feature in the UI.
- or-
- Create a new REG\_DWORD registry setting named **HttpAcceptLanguageOptOut** in **HKEY\_CURRENT\_USER\Control Panel\International\User Profile** with a value of 1.

To turn off **Let Windows track app launches to improve Start and search results**:

- Turn off the feature in the UI.
- or-
- Create a REG\_DWORD registry setting named **Start\_TrackProgs** in **HKEY\_CURRENT\_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Advanced** with value of 0 (zero).

### Windows Server 2016 and Windows 10, version 1607 and earlier options

To turn off **Let apps use my advertising ID for experiences across apps (turning this off will reset your**

ID):

**NOTE**

When you turn this feature off in the UI, it turns off the advertising ID, not just resets it.

- Turn off the feature in the UI.  
-or-
- Apply the Group Policy: **Computer Configuration > Administrative Templates > System > User Profiles > Turn off the advertising ID.**  
-or-
- Create a REG\_DWORD registry setting named **Enabled** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\AdvertisingInfo** with a value of 0 (zero).  
-or-
- Create a REG\_DWORD registry setting named **DisabledByGroupPolicy** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\Windows\AdvertisingInfo** with a value of 1 (one).

To turn off **Turn on SmartScreen Filter to check web content (URLs) that Microsoft Store apps use:**

- Turn off the feature in the UI.  
-or-
- In Windows Server 2016, apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Microsoft Edge > Configure SmartScreen Filter.** In Windows 10, version 1703, apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Microsoft Edge > Configure Windows Defender SmartScreen Filter.**  
  
In Windows Server 2016, apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > File Explorer > Configure Windows SmartScreen.** In Windows 10, version 1703, apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > File Explorer > Configure Windows Defender SmartScreen.**  
-or-
- Apply the Browser/AllowSmartScreen MDM policy from the [Policy CSP](#) where 0 is turned off and 1 is turned on.  
-or-
- Create a provisioning package, using:
  - For Internet Explorer: **Runtime settings > Policies > Browser > AllowSmartScreen**
  - For Microsoft Edge: **Runtime settings > Policies > MicrosoftEdge > AllowSmartScreen**-or-
- Create a REG\_DWORD registry setting named **EnableWebContentEvaluation** in **HKEY\_CURRENT\_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\AppHost** with a value

of 0 (zero).

-or-

- Create a REG\_DWORD registry setting named **EnableSmartScreen** in **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\Windows\System** with a value of 0 (zero).

To turn off **Send Microsoft info about how I write to help us improve typing and writing in the future**:

#### NOTE

If the diagnostic data level is set to either **Basic** or **Security**, this is turned off automatically.

- Turn off the feature in the UI.

-or-

- Apply the TextInput/AllowLinguisticDataCollection MDM policy from the [Policy CSP](#) where:
  - **0**. Not allowed
  - **1**. Allowed (default)

To turn off **Let websites provide locally relevant content by accessing my language list**:

- Turn off the feature in the UI.

-or-

- Create a new REG\_DWORD registry setting named **HttpAcceptLanguageOptOut** in **HKEY\_CURRENT\_USER\Control Panel\International\User Profile** with a value of 1.

To turn off **Let apps on my other devices open apps and continue experiences on this devices**:

- Turn off the feature in the UI.

-or-

- Disable the Group Policy: **Computer Configuration > Administrative Templates > System > Group Policy > Continue experiences on this device**.

-or-

- Create a REG\_DWORD registry setting named **EnableCdp** in **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\Windows\System** with a value of 0 (zero).

To turn off **Let apps on my other devices use Bluetooth to open apps and continue experiences on this device**:

- Turn off the feature in the UI.

## 17.2 Location

In the **Location** area, you choose whether devices have access to location-specific sensors and which apps have access to the device's location.

To turn off **Location for this device**:

- Click the **Change** button in the UI.

-or-

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows**



## Components > Location and Sensors > Turn off location.

-or-

- Create a REG\_DWORD registry setting named **LetAppsAccessLocation** in **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\Windows\AppPrivacy** with a value of 2 (two).

-or-

- Apply the System/AllowLocation MDM policy from the [Policy CSP](#), where:
  - **0.** Turned off and the employee can't turn it back on.
  - **1.** Turned on, but lets the employee choose whether to use it. (default)
  - **2.** Turned on and the employee can't turn it off.

### NOTE

You can also set this MDM policy in System Center Configuration Manager using the [WMI Bridge Provider](#).

-or-

- Create a provisioning package, using **Runtime settings > Policies > System > AllowLocation**, where
  - **No.** Turns off location service.
  - **Yes.** Turns on location service. (default)

To turn off **Location**:

- Turn off the feature in the UI.
- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access location**
  - Set the **Select a setting** box to **Force Deny**.

-or-

- Create a REG\_DWORD registry setting named **DisableLocation** in **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\Windows\LocationAndSensors** with a value of 1 (one).

-or-

To turn off **Location history**:

- Erase the history using the **Clear** button in the UI.

To turn off **Choose apps that can use your location**:

- Turn off each app using the UI.

## 17.3 Camera

In the **Camera** area, you can choose which apps can access a device's camera.

To turn off **Let apps use my camera**:

- Turn off the feature in the UI.

-or-

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access the camera**
  - Set the **Select a setting** box to **Force Deny**.

-or-

- Create a REG\_DWORD registry setting named **LetAppsAccessCamera** in **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\Windows\AppPrivacy** with a value of 2 (two).

-or-

- Apply the Camera/AllowCamera MDM policy from the [Policy CSP](#), where:
  - **0**. Apps can't use the camera.
  - **1**. Apps can use the camera.

#### NOTE

You can also set this MDM policy in System Center Configuration Manager using the [WMI Bridge Provider](#).

-or-

- Create a provisioning package with use Windows ICD, using **Runtime settings > Policies > Camera > AllowCamera**, where:
  - **0**. Apps can't use the camera.
  - **1**. Apps can use the camera.

To turn off **Choose apps that can use your camera**:

- Turn off the feature in the UI for each app.

## 17.4 Microphone

In the **Microphone** area, you can choose which apps can access a device's microphone.

To turn off **Let apps use my microphone**:

- Turn off the feature in the UI.

-or-

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access the microphone**
  - Set the **Select a setting** box to **Force Deny**.

-or-

- Apply the Privacy/LetAppsAccessMicrophone MDM policy from the [Policy CSP](#), where:
  - **0**. User in control
  - **1**. Force allow
  - **2**. Force deny

-or-

- Create a REG\_DWORD registry setting named **LetAppsAccessMicrophone** in **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\Windows\AppPrivacy** with a value of 2 (two)

To turn off **Choose apps that can use your microphone**:

- Turn off the feature in the UI for each app.

## 17.5 Notifications

### IMPORTANT

Disabling notifications will also disable the ability to manage the device through MDM. If you are using an MDM solution, make sure cloud notifications are enabled through one of the options below.

To turn off notifications network usage:

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Start Menu and Taskbar > Notifications > Turn off Notifications network usage**

- Set to **Enabled**.

-or-

- Create a REG\_DWORD registry setting named **NoCloudApplicationNotification** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\Windows\CurrentVersion\PushNotifications** with a value of 1 (one)

-or-

- Apply the Notifications/DisallowCloudNotification MDM policy from the [Policy CSP](#), where:
  - **0**. WNS notifications allowed
  - **1**. No WNS notifications allowed

In the **Notifications** area, you can also choose which apps have access to notifications.

To turn off **Let apps access my notifications**:

- Turn off the feature in the UI.

-or-

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access notifications**

- Set the **Select a setting** box to **Force Deny**.

-or-

- Apply the Privacy/LetAppsAccessNotifications MDM policy from the [Policy CSP](#), where:
  - **0**. User in control
  - **1**. Force allow
  - **2**. Force deny

-or-

- Create a REG\_DWORD registry setting named **LetAppsAccessNotifications** in **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\Windows\AppPrivacy** with a value of 2 (two)

## 17.6 Speech, inking, & typing

In the **Speech, Inking, & Typing** area, you can let Windows and Cortana better understand your employee's voice and written input by sampling their voice and writing, and by comparing verbal and written input to contact names and calendar entrees.

### NOTE

For more info on how to disable Cortana in your enterprise, see [Cortana](#) in this article.

To turn off the functionality:

- Click the **Stop getting to know me** button, and then click **Turn off**.

-or-

- Enable the Group Policy: **Computer Configuration > Administrative Templates > Control Panel > Regional and Language Options > Handwriting personalization > Turn off automatic learning**

-or-

- Create a REG\_DWORD registry setting named **RestrictImplicitInkCollection** in **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\InputPersonalization** with a value of 1 (one).

-or-

- Create a REG\_DWORD registry setting named **AcceptedPrivacyPolicy** in **HKEY\_CURRENT\_USER\Software\Microsoft\Personalization\Settings** with a value of 0 (zero).

-and-

- Create a REG\_DWORD registry setting named **HarvestContacts** in **HKEY\_CURRENT\_USER\Software\Microsoft\InputPersonalization\TrainedDataStore** with a value of 0 (zero).

If you're running at least Windows 10, version 1703, you can turn off updates to the speech recognition and speech synthesis models:

- Disable the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Speech > Allow automatically update of Speech Data**

If you're running at least Windows 10, version 1607, you can turn off updates to the speech recognition and speech synthesis models:

Apply the Speech/AllowSpeechModelUpdate MDM policy from the [Policy CSP](#), where:

- **0** (default). Not allowed.
- **1**. Allowed.

-or-

- Create a REG\_DWORD registry setting named **ModelDownloadAllowed** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Speech\_OneCore\Preferences** with a value of 0 (zero).

## 17.7 Account info

In the **Account Info** area, you can choose which apps can access your name, picture, and other account info.

To turn off **Let apps access my name, picture, and other account info**:

- Turn off the feature in the UI.

-or-

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access account information**
  - Set the **Select a setting** box to **Force Deny**.

-or-

- Apply the Privacy/LetAppsAccessAccountInfo MDM policy from the [Policy CSP](#), where:
  - **0.** User in control
  - **1.** Force allow
  - **2.** Force deny

-or-

- Create a REG\_DWORD registry setting named **LetAppsAccessAccountInfo** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\AppPrivacy** with a value of 2 (two).

To turn off **Choose the apps that can access your account info**:

- Turn off the feature in the UI for each app.

### 17.8 Contacts

In the **Contacts** area, you can choose which apps can access an employee's contacts list.

To turn off **Choose apps that can access contacts**:

- Turn off the feature in the UI for each app.

-or-

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access contacts**
  - Set the **Select a setting** box to **Force Deny**.

-or-

- Apply the Privacy/LetAppsAccessContacts MDM policy from the [Policy CSP](#), where:
  - **0.** User in control
  - **1.** Force allow
  - **2.** Force deny

-or-

- Create a REG\_DWORD registry setting named **LetAppsAccessContacts** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\AppPrivacy** with a value of 2 (two).

### 17.9 Calendar

In the **Calendar** area, you can choose which apps have access to an employee's calendar.

To turn off **Let apps access my calendar**:

- Turn off the feature in the UI.

-or-

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access the calendar**
  - Set the **Select a setting** box to **Force Deny**.

-or-

- Apply the Privacy/LetAppsAccessCalendar MDM policy from the [Policy CSP](#), where:
  - **0**. User in control
  - **1**. Force allow
  - **2**. Force deny

-or-

- Create a REG\_DWORD registry setting named **LetAppsAccessCalendar** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\AppPrivacy** with a value of 2 (two).

To turn off **Choose apps that can access calendar**:

- Turn off the feature in the UI for each app.

### 17.10 Call history

In the **Call history** area, you can choose which apps have access to an employee's call history.

To turn off **Let apps access my call history**:

- Turn off the feature in the UI.

-or-

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access call history**
  - Set the **Select a setting** box to **Force Deny**.

-or-

- Apply the Privacy/LetAppsAccessCallHistory MDM policy from the [Policy CSP](#), where:
  - **0**. User in control
  - **1**. Force allow
  - **2**. Force deny

-or-

- Create a REG\_DWORD registry setting named **LetAppsAccessCallHistory** in **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\Windows\AppPrivacy** with a value of 2 (two).

### 17.11 Email

In the **Email** area, you can choose which apps have can access and send email.

To turn off **Let apps access and send email**:

- Turn off the feature in the UI.

-or-

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access email**

- Set the **Select a setting** box to **Force Deny**.

-or-

- Apply the Privacy/LetAppsAccessEmail MDM policy from the [Policy CSP](#), where:
  - **0**. User in control
  - **1**. Force allow
  - **2**. Force deny

-or-

- Create a REG\_DWORD registry setting named **LetAppsAccessEmail** in **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\Windows\AppPrivacy** with a value of 2 (two).

## 17.12 Messaging

In the **Messaging** area, you can choose which apps can read or send messages.

To turn off **Let apps read or send messages (text or MMS)**:

- Turn off the feature in the UI.

-or-

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access messaging**

- Set the **Select a setting** box to **Force Deny**.

-or-

- Apply the Privacy/LetAppsAccessMessaging MDM policy from the [Policy CSP](#), where:
  - **0**. User in control
  - **1**. Force allow
  - **2**. Force deny

-or-

- Create a REG\_DWORD registry setting named **LetAppsAccessMessaging** in **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\Windows\AppPrivacy** with a value of 2 (two).

To turn off **Choose apps that can read or send messages**:

- Turn off the feature in the UI for each app.

## 17.13 Phone calls

In the **Phone calls** area, you can choose which apps can make phone calls.

To turn off **Let apps make phone calls**:

- Turn off the feature in the UI.

-or-

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps make phone calls**

- Set the **Select a setting** box to **Force Deny**.

-or-

- Apply the Privacy/LetAppsAccessPhone MDM policy from the [Policy CSP](#), where:
  - **0.** User in control
  - **1.** Force allow
  - **2.** Force deny

-or-

- Create a REG\_DWORD registry setting named **LetAppsAccessPhone** in **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\Windows\AppPrivacy** with a value of 2 (two).

To turn off **Choose apps that can make phone calls**:

- Turn off the feature in the UI for each app.

#### 17.14 Radios

In the **Radios** area, you can choose which apps can turn a device's radio on or off.

To turn off **Let apps control radios**:

- Turn off the feature in the UI.

-or-

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps control radios**

- Set the **Select a setting** box to **Force Deny**.

-or-

- Apply the Privacy/LetAppsAccessRadios MDM policy from the [Policy CSP](#), where:
  - **0.** User in control
  - **1.** Force allow
  - **2.** Force deny

-or-

- Create a REG\_DWORD registry setting named **LetAppsAccessRadios** in **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\Windows\AppPrivacy** with a value of 2 (two).

To turn off **Choose apps that can control radios**:

- Turn off the feature in the UI for each app.

#### 17.15 Other devices



In the **Other Devices** area, you can choose whether devices that aren't paired to PCs, such as an Xbox One, can share and sync info.

To turn off **Let apps automatically share and sync info with wireless devices that don't explicitly pair with your PC, tablet, or phone**:

- Turn off the feature in the UI.  
-or-
- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps sync with devices**  
-or-
- Apply the Privacy/LetAppsSyncWithDevices MDM policy from the [Policy CSP](#), where:
  - **0.** User in control
  - **1.** Force allow
  - **2.** Force deny-or-
- Create a REG\_DWORD registry setting named **LetAppsSyncWithDevices** in **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\Windows\AppPrivacy** with a value of 2 (two).

To turn off **Let your apps use your trusted devices (hardware you've already connected, or comes with your PC, tablet, or phone)**:

- Turn off the feature in the UI.  
-or-
- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access trusted devices**
  - Set the **Select a setting** box to **Force Deny**.-or-
- Apply the **Privacy/LetAppsAccessTrustedDevices** MDM policy from the [Policy CSP](#), where:
  - **0.** User in control
  - **1.** Force allow
  - **2.** Force deny

### 17.16 Feedback & diagnostics

In the **Feedback & Diagnostics** area, you can choose how often you're asked for feedback and how much diagnostic and usage information is sent to Microsoft.

To change how frequently **Windows should ask for my feedback**:

#### NOTE

Feedback frequency only applies to user-generated feedback, not diagnostic and usage data sent from the device.

- To change from **Automatically (Recommended)**, use the drop-down list in the UI.

-or-

- Enable the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Data Collection and Preview Builds > Do not show feedback notifications**

-or-

- Create a REG\_DWORD registry setting named **DoNotShowFeedbackNotifications** in **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\Windows\DataCollection** with a value of 1 (one).

-or-

- Create the registry keys (REG\_DWORD type):
  - HKEY\_CURRENT\_USER\Software\Microsoft\Siuf\Rules\PeriodInNanoSeconds
  - HKEY\_CURRENT\_USER\Software\Microsoft\Siuf\Rules\NumberOfSIUFInPeriod

Based on these settings:

SETTING	PERIODINNANOSECONDS	NUMBEROFSIUFINPERIOD
Automatically	Delete the registry setting	Delete the registry setting
Never	0	0
Always	100000000	Delete the registry setting
Once a day	864000000000	1
Once a week	6048000000000	1

To change the level of diagnostic and usage data sent when you **Send your device data to Microsoft**:

- Click either the **Basic** or **Full** options.

-or-

- Apply the Group Policy: **Computer Configuration\Administrative Templates\Windows Components\Data Collection And Preview Builds\Allow Telemetry** and select the appropriate option for your deployment.

-or-

- Create a REG\_DWORD registry setting in **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\Windows\DataCollection\AllowTelemetry** with a value of 0-3, as appropriate for your deployment (see below for the values for each level).

#### NOTE

If the **Security** option is configured by using Group Policy or the Registry, the value will not be reflected in the UI. The **Security** option is only available in Windows 10 Enterprise edition.

-or-

- Apply the System/AllowTelemetry MDM policy from the [Policy CSP](#), where:
  - **0**. Maps to the **Security** level.

- 1. Maps to the **Basic** level.
- 2. Maps to the **Enhanced** level.
- 3. Maps to the **Full** level.

-or-

- Create a provisioning package, using **Runtime settings > Policies > System > AllowTelemetry**, where:
  - 0. Maps to the **Security** level.
  - 1. Maps to the **Basic** level.
  - 2. Maps to the **Enhanced** level.
  - 3. Maps to the **Full** level.

To turn off tailored experiences with relevant tips and recommendations by using your diagnostics data:

- Turn off the feature in the UI.

-or-

- Apply the Group Policy: **User Configuration > Administrative Templates > Windows Components > Cloud Content > Do not use diagnostic data for tailored experiences**

### 17.17 Background apps

In the **Background Apps** area, you can choose which apps can run in the background.

To turn off **Let apps run in the background**:

- In **Background apps**, set **Let apps run in the background** to **Off**.

-or-

- In **Background apps**, turn off the feature for each app.

-or-

- Apply the Group Policy (only applicable for Windows 10, version 1703): **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps run in the background**

- Set the **Select a setting** box to **Force Deny**.

-or-

- Apply the Privacy/LetAppsRunInBackground MDM policy from the [Policy CSP](#), where:
  - 0. User in control
  - 1. Force allow
  - 2. Force deny

#### NOTE

Some apps, including Cortana and Search, might not function as expected if you set **Let apps run in the background** to **Force Deny**.

### 17.18 Motion

In the **Motion** area, you can choose which apps have access to your motion data.

To turn off **Let Windows and your apps use your motion data and collect motion history**:

- Turn off the feature in the UI.

-or-

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access motion**

-or-

- Apply the Privacy/LetAppsAccessMotion MDM policy from the [Policy CSP](#), where:
- **0.** User in control
- **1.** Force allow
- **2.** Force deny

-or-

- Create a REG\_DWORD registry setting named **LetAppsAccessMotion** in **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\Windows\AppPrivacy** with a value of 2 (two).

### 17.19 Tasks

In the **Tasks** area, you can choose which apps have access to your tasks.

To turn this off:

- Turn off the feature in the UI.

-or-

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access Tasks**
  - Set the **Select a setting** box to **Force Deny**.

-or-

- Apply the Privacy/LetAppsAccessTasks MDM policy from the [Policy CSP](#), where:
- **0.** User in control
- **1.** Force allow
- **2.** Force deny

### 17.20 App Diagnostics

In the **App diagnostics** area, you can choose which apps have access to your diagnostic information.

To turn this off:

- Turn off the feature in the UI.

-or-

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access diagnostic information about other apps**

-or-

- Apply the Privacy/LetAppsGetDiagnosticInfo MDM policy from the [Policy CSP](#), where:

- 0. User in control
- 1. Force allow
- 2. Force deny

## 18. Software Protection Platform

Enterprise customers can manage their Windows activation status with volume licensing using an on-premises Key Management Server. You can opt out of sending KMS client activation data to Microsoft automatically by doing one of the following:

For Windows 10:

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Software Protection Platform > Turn off KMS Client Online AVS Validation**
- or-
- Apply the Licensing/DisallowKMSClientOnlineAVSValidation MDM policy from the [Policy CSP](#) where 0 is disabled (default) and 1 is enabled.

-or-

- Create a REG\_DWORD registry setting named **NoGenTicket** in **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\Windows NT\CurrentVersion\Software Protection Platform** with a value of 1 (one).

For Windows Server 2016 with Desktop Experience or Windows Server 2016 Server Core:

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Software Protection Platform > Turn off KMS Client Online AVS Validation**
- or-
- Create a REG\_DWORD registry setting named **NoGenTicket** in **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\Windows NT\CurrentVersion\Software Protection Platform** with a value of 1 (one).

The Windows activation status will be valid for a rolling period of 180 days with weekly activation status checks to the KMS.

## 19. Storage health

Enterprise customers can manage updates to the Disk Failure Prediction Model.

For Windows 10:

- Apply the Group Policy: **Computer Configuration > Administrative Templates > System > Storage Health > Allow downloading updates to the Disk Failure Prediction Model**

## 20. Sync your settings

You can control if your settings are synchronized:

- In the UI: **Settings > Accounts > Sync your settings**
- or-
- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Sync your settings > Do not sync**

-or-

- Create a REG\_DWORD registry setting named **DisableSettingSync** in **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\Windows\SettingSync** with a value of 2 (two) and another named **DisableSettingSyncUserOverride** in **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\Windows\SettingSync** with a value of 1 (one).

-or-

- Apply the Experience/AllowSyncMySettings MDM policy from the [Policy CSP](#) where 0 is not allowed and 1 is allowed.

-or-

- Create a provisioning package, using **Runtime settings > Policies > Experience > AllowSyncMySettings**, where
  - **No.** Settings are not synchronized.
  - **Yes.** Settings are synchronized. (default)

To turn off Messaging cloud sync:

- Create a REG\_DWORD registry setting named **CloudServiceSyncEnabled** in **HKEY\_CURRENT\_USER\SOFTWARE\Microsoft\Messaging** with a value of 0 (zero).

## 21. Teredo

You can disable Teredo by using Group Policy or by using the netsh.exe command. For more info on Teredo, see [Internet Protocol Version 6, Teredo, and Related Technologies](#).

### NOTE

If you disable Teredo, some XBOX gaming features and Windows Update Delivery Optimization will not work.

- Enable the Group Policy: **Computer Configuration > Administrative Templates > Network > TCPIP Settings > IPv6 Transition Technologies > Set Teredo State** and set it to **Disabled State**.

-or-

- Create a new REG\_SZ registry setting named **Teredo\_State** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\Windows\TCPIP\v6Transition** with a value of **Disabled**.

-or-

- From an elevated command prompt, run **netsh interface teredo set state disabled**

## 22. Wi-Fi Sense

### IMPORTANT

Beginning with Windows 10, version 1803, Wi-Fi Sense is no longer available. The following section only applies to Windows 10, version 1709 and prior. Please see [Connecting to open Wi-Fi hotspots in Windows 10](#) for more details.

Wi-Fi Sense automatically connects devices to known hotspots and to the wireless networks the person's contacts have shared with them.

To turn off **Connect to suggested open hotspots** and **Connect to networks shared by my contacts**:

- Turn off the feature in the UI.

-or-

- Disable the Group Policy: **Computer Configuration > Administrative Templates > Network > WLAN Service > WLAN Settings > Allow Windows to automatically connect to suggested open hotspots, to networks shared by contacts, and to hotspots offering paid services.**

-or-

- Create a new REG\_DWORD registry setting named **AutoConnectAllowedOEM** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\WcmSvc\wifinetworkmanager\config** with a value of 0 (zero).

-or-

- Change the Windows Provisioning setting, `WiFiSenseAllowed`, to 0 (zero). For more info, see the Windows Provisioning Settings reference doc, [WiFiSenseAllowed](#).

-or-

- Use the Unattended settings to set the value of `WiFiSenseAllowed` to 0 (zero). For more info, see the Unattended Windows Setup reference doc, [WiFiSenseAllowed](#).

When turned off, the Wi-Fi Sense settings still appear on the Wi-Fi Settings screen, but they're non-functional and they can't be controlled by the employee.

### 23. Windows Defender

You can disconnect from the Microsoft Antimalware Protection Service.

- Disable the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Windows Defender Antivirus > MAPS > Join Microsoft MAPS**

-or-

- Delete the registry setting **named** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\Windows Defender\Updates.**

-or-

- For Windows 10 only, apply the Defender/AllowCloudProtection MDM policy from the [Defender CSP](#).

-or-

- Use the registry to set the REG\_DWORD value **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\Windows Defender\Spynet\SpyNetReporting** to 0 (zero).

-and-

From an elevated Windows PowerShell prompt, run **set-mppreference -Mapsreporting 0**

You can stop sending file samples back to Microsoft.

- Set the Group Policy **Computer Configuration > Administrative Templates > Windows Components > Windows Defender Antivirus > MAPS > Send file samples when further analysis is required** to **Always Prompt** or **Never Send**.

-or-

- For Windows 10 only, apply the Defender/SubmitSamplesConsent MDM policy from the [Policy CSP](#), where:
  - **0**. Always prompt.

- 1. (default) Send safe samples automatically.
- 2. Never send.
- 3. Send all samples automatically.

-or-

- Use the registry to set the REG\_DWORD value **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\Windows Defender\Spynet\SubmitSamplesConsent** to 0 (zero) to always prompt or 2 to never send.

You can stop downloading definition updates:

- Enable the Group Policy **Computer Configuration > Administrative Templates > Windows Components > Windows Defender Antivirus > Signature Updates > Define the order of sources for downloading definition updates** and set it to **FileShares**.

-and-

- Disable the Group Policy **Computer Configuration > Administrative Templates > Windows Components > Windows Defender Antivirus > Signature Updates > Define file shares for downloading definition updates** and set it to nothing.

-or-

- Create a new REG\_SZ registry setting named **FallbackOrder** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\Windows Defender\Updates** with a value of **FileShares**.

For Windows 10 only, you can stop Enhanced Notifications:

- Turn off the feature in the UI.

You can also use the registry to turn off Malicious Software Reporting Tool diagnostic data by setting the REG\_DWORD value

**HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\MRT\DontReportInfectionInformation** to 1.

## 24. Windows Media Player

To remove Windows Media Player on Windows 10:

- From the **Programs and Features** control panel, click **Turn Windows features on or off**, under **Media Features**, clear the **Windows Media Player** check box, and then click **OK**.

-or-

- Run the following DISM command from an elevated command prompt: **dism /online /Disable-Feature /FeatureName:WindowsMediaPlayer**

To remove Windows Media Player on Windows Server 2016:

- Run the following DISM command from an elevated command prompt: **dism /online /Disable-Feature /FeatureName:WindowsMediaPlayer**

## 25. Windows Spotlight

Windows Spotlight provides features such as different background images and text on the lock screen, suggested apps, Microsoft account notifications, and Windows tips. You can control it by using the user interface, MDM policy, or through Group Policy.

If you're running Windows 10, version 1607 or later, you only need to enable the following Group Policy:



- **User Configuration > Administrative Templates > Windows Components > Cloud Content > Turn off all Windows spotlight features**

**NOTE**

This must be done within 15 minutes after Windows 10 is installed. Alternatively, you can create an image with this setting.

-or-

- For Windows 10 only, apply the Experience/AllowWindowsSpotlight MDM policy from the [Policy CSP](#), with a value of 0 (zero).

-or-

- Create a new REG\_DWORD registry setting named **DisableWindowsSpotlightFeatures** in **HKEY\_CURRENT\_USER\SOFTWARE\Policies\Microsoft\Windows\CloudContent** with a value of 1 (one).

If you're not running Windows 10, version 1607 or later, you can use the other options in this section.

- Configure the following in **Settings**:
  - **Personalization > Lock screen > Background > Windows spotlight**, select a different background, and turn off **Get fun facts, tips, tricks and more on your lock screen**.

**NOTE**

In Windows 10, version 1507 and Windows 10, version 1511, this setting was named **Show me tips, tricks, and more on the lock screen**.

- **Personalization > Start > Occasionally show suggestions in Start**.
- **System > Notifications & actions > Show me tips about Windows**.

-or-

- Apply the Group Policies:
  - **Computer Configuration > Administrative Templates > Control Panel > Personalization > Force a specific default lock screen image**.
    - Add a location in the **Path to local lock screen image** box.
    - Set the **Turn off fun facts, tips, tricks, and more on lock screen** check box.

#### NOTE

This will only take effect if the policy is applied before the first logon. If you cannot apply the **Force a specific default lock screen image** policy before the first logon to the device, you can apply this policy: **Computer Configuration > Administrative Templates > Control Panel > Personalization > Do not display the lock screen**. Alternatively, you can create a new REG\_SZ registry setting named **LockScreenImage** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Personalization** with a value of **C:\windows\web\screen\lockscreen.jpg** and create a new REG\_DWORD registry setting named **LockScreenOverlaysDisabled** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Personalization** with a value of 1 (one).

- **Computer Configuration > Administrative Templates > Windows Components > Cloud Content > Do not show Windows tips.**

-or-

- Create a new REG\_DWORD registry setting named **DisableSoftLanding** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\Windows\CloudContent** with a value of 1 (one).
- **Computer Configuration > Administrative Templates > Windows Components > Cloud Content > Turn off Microsoft consumer experiences.**

-or-

- Create a new REG\_DWORD registry setting named **DisableWindowsConsumerFeatures** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\Windows\CloudContent** with a value of 1 (one).

For more info, see [Windows Spotlight on the lock screen](#).

## 26. Microsoft Store

You can turn off the ability to launch apps from the Microsoft Store that were preinstalled or downloaded. This will also turn off automatic app updates, and the Microsoft Store will be disabled. In addition, new email accounts cannot be created by clicking **Settings > Accounts > Email & app accounts > Add an account**. On Windows Server 2016, this will block Microsoft Store calls from Universal Windows Apps.

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Store > Disable all apps from Microsoft Store.**

-or-

- Create a new REG\_DWORD registry setting named **DisableStoreApps** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\WindowsStore** with a value of 1 (one).
- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Store > Turn off Automatic Download and Install of updates.**

-or-

- Create a new REG\_DWORD registry setting named **AutoDownload** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\WindowsStore** with a value of 2 (two).

### 26.1 Apps for websites

You can turn off apps for websites, preventing customers who visit websites that are registered with their associated app from directly launching the app.

Disable the Group Policy: **Computer Configuration > Administrative Templates > System > Group Policy > Configure web-to-app linking with URI handlers**

## 27. Windows Update Delivery Optimization

Windows Update Delivery Optimization lets you get Windows updates and Microsoft Store apps from sources in addition to Microsoft, which not only helps when you have a limited or unreliable Internet connection, but can also help you reduce the amount of bandwidth needed to keep all of your organization's PCs up-to-date. If you have Delivery Optimization turned on, PCs on your network may send and receive updates and apps to other PCs on your local network, if you choose, or to PCs on the Internet.

By default, PCs running Windows 10 Enterprise and Windows 10 Education will only use Delivery Optimization to get and receive updates for PCs and apps on your local network.

Use the UI, Group Policy, MDM policies, or Windows Provisioning to set up Delivery Optimization.

In Windows 10, version 1607, you can stop network traffic related to Windows Update Delivery Optimization by setting **Download Mode** to **Simple** (99) or **Bypass** (100), as described below.

### 27.1 Settings > Update & security

You can set up Delivery Optimization from the **Settings** UI.

- Go to **Settings > Update & security > Windows Update > Advanced options > Choose how updates are delivered**.

### 27.2 Delivery Optimization Group Policies

You can find the Delivery Optimization Group Policy objects under **Computer Configuration > Administrative Templates > Windows Components > Delivery Optimization**.

POLICY	DESCRIPTION
Download Mode	Lets you choose where Delivery Optimization gets or sends updates and apps, including <ul style="list-style-type: none"><li>• <b>None</b>. Turns off Delivery Optimization.</li><li>• <b>Group</b>. Gets or sends updates and apps to PCs on the same local network domain.</li><li>• <b>Internet</b>. Gets or sends updates and apps to PCs on the Internet.</li><li>• <b>LAN</b>. Gets or sends updates and apps to PCs on the same NAT only.</li><li>• <b>Simple</b>. Simple download mode with no peering.</li><li>• <b>Bypass</b>. Use BITS instead of Windows Update Delivery Optimization.</li></ul>
Group ID	Lets you provide a Group ID that limits which PCs can share apps and updates. <b>Note:</b> This ID must be a GUID.
Max Cache Age	Lets you specify the maximum time (in seconds) that a file is held in the Delivery Optimization cache. The default value is 259200 seconds (3 days).

POLICY	DESCRIPTION
Max Cache Size	Lets you specify the maximum cache size as a percentage of disk size. The default value is 20, which represents 20% of the disk.
Max Upload Bandwidth	Lets you specify the maximum upload bandwidth (in KB/second) that a device uses across all concurrent upload activity. The default value is 0, which means unlimited possible bandwidth.

You can also set the **Download Mode** policy by creating a new REG\_DWORD registry setting named **DODownloadMode** in **HKEY\_LOCAL\_MACHINE\SOFTWARE\Policies\Microsoft\Windows\DeliveryOptimization** with a value of 100 (one hundred).

### 27.3 Delivery Optimization MDM policies

The following Delivery Optimization MDM policies are available in the [Policy CSP](#).

POLICY	DESCRIPTION
DeliveryOptimization/DODownloadMode	Lets you choose where Delivery Optimization gets or sends updates and apps, including <ul style="list-style-type: none"> <li>• <b>0</b>. Turns off Delivery Optimization.</li> <li>• <b>1</b>. Gets or sends updates and apps to PCs on the same NAT only.</li> <li>• <b>2</b>. Gets or sends updates and apps to PCs on the same local network domain.</li> <li>• <b>3</b>. Gets or sends updates and apps to PCs on the Internet.</li> <li>• <b>99</b>. Simple download mode with no peering.</li> <li>• <b>100</b>. Use BITS instead of Windows Update Delivery Optimization.</li> </ul>
DeliveryOptimization/DOGroupID	Lets you provide a Group ID that limits which PCs can share apps and updates. <b>Note</b> This ID must be a GUID.
DeliveryOptimization/DOMaxCacheAge	Lets you specify the maximum time (in seconds) that a file is held in the Delivery Optimization cache. The default value is 259200 seconds (3 days).
DeliveryOptimization/DOMaxCacheSize	Lets you specify the maximum cache size as a percentage of disk size. The default value is 20, which represents 20% of the disk.
DeliveryOptimization/DOMaxUploadBandwidth	Lets you specify the maximum upload bandwidth (in KB/second) that a device uses across all concurrent upload activity. The default value is 0, which means unlimited possible bandwidth.

### 27.4 Delivery Optimization Windows Provisioning

If you don't have an MDM server in your enterprise, you can use Windows Provisioning to configure the Delivery Optimization policies

Use Windows ICD, included with the [Windows Assessment and Deployment Kit \(Windows ADK\)](#), to create a provisioning package for Delivery Optimization.

1. Open Windows ICD, and then click **New provisioning package**.
2. In the **Name** box, type a name for the provisioning package, and then click **Next**.
3. Click the **Common to all Windows editions** option, click **Next**, and then click **Finish**.
4. Go to **Runtime settings > Policies > Delivery Optimization** to configure the policies.

For more info about Delivery Optimization in general, see [Windows Update Delivery Optimization: FAQ](#).

## 28. Windows Update

You can turn off Windows Update by setting the following registry entries:

- Add a REG\_DWORD value named **DoNotConnectToWindowsUpdateInternetLocations** to **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\Windows\WindowsUpdate** and set the value to 1.  
  
-and-
- Add a REG\_DWORD value named **DisableWindowsUpdateAccess** to **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\Windows\WindowsUpdate** and set the value to 1.  
  
-and-
- Add a REG\_DWORD value named **UseWUServer** to **HKEY\_LOCAL\_MACHINE\Software\Policies\Microsoft\Windows\WindowsUpdate\AU** and set the value to 1.  
  
-or-
- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Windows Update > Do not connect to any Windows Update Internet locations**.  
  
-and-
- Apply the Group Policy: **Computer Configuration > Administrative Templates > System > Internet Communication Management > Internet Communication Settings > Turn off access to all Windows Update features**.  
  
-and-
- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Windows Update > Specify intranet Microsoft update service location** and set the **Set the alternate download server** to " ".

You can turn off automatic updates by doing one of the following. This is not recommended.

- Add a REG\_DWORD value named **AutoDownload** to **HKEY\_LOCAL\_MACHINE\Software\Microsoft\Windows\CurrentVersion\WindowsStore\WindowsUpdate** and set the value to 5.  
  
-or-
- For Windows 10 only, apply the Update/AllowAutoUpdate MDM policy from the [Policy CSP](#), where:

- **0.** Notify the user before downloading the update.
- **1.** Auto install the update and then notify the user to schedule a device restart.
- **2** (default). Auto install and restart.
- **3.** Auto install and restart at a specified time.
- **4.** Auto install and restart without end-user control.
- **5.** Turn off automatic updates.

To learn more, see [Device update management](#) and [Configure Automatic Updates by using Group Policy](#).