

Collecting Information About Computers

Cmdlets from **CimCmdlets** module are the most important cmdlets for general system management tasks. All critical subsystem settings are exposed through WMI. Furthermore, WMI treats data as objects that are in collections of one or more items. Because Windows PowerShell also works with objects and has a pipeline that allows you to treat single or multiple objects in the same way, generic WMI access allows you to perform some advanced tasks with very little work.

Listing Desktop Settings

We'll begin with a command that collects information about the desktops on the local computer.

PowerShellCopy

```
Get-CimInstance -ClassName Win32_Desktop
```

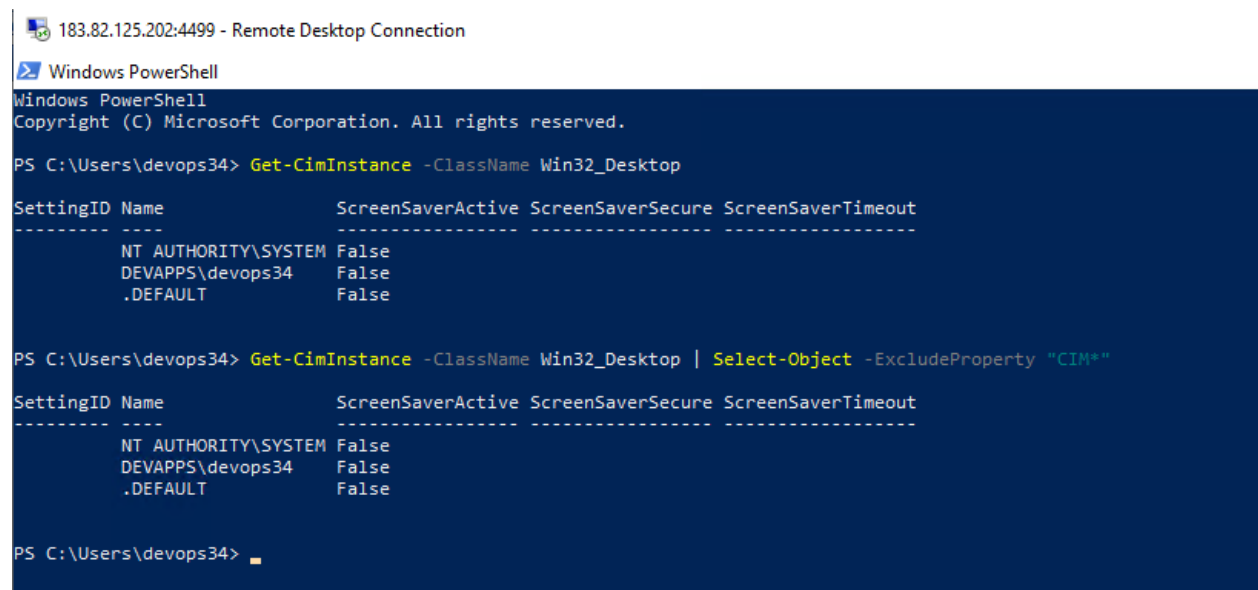
Because most of these metadata properties have names that begin with **Cim**, you can filter the properties using `Select-Object`. Specify the **-ExcludeProperty** parameter with "Cim*" as the value. For example:

PowerShellCopy

```
Get-CimInstance -ClassName Win32_Desktop | Select-Object -ExcludeProperty "Cim*"
```

To filter out the metadata, use a pipeline operator (|) to send the results of the `Get-CimInstance` command to `Select-Object -ExcludeProperty "Cim*"`.

Output :



```
183.82.125.202:4499 - Remote Desktop Connection
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\devops34> Get-CimInstance -ClassName Win32_Desktop

SettingID Name                ScreenSaverActive ScreenSaverSecure ScreenSaverTimeout
-----
NT AUTHORITY\SYSTEM False
DEVAPPS\devops34 False
.DEFAULT False

PS C:\Users\devops34> Get-CimInstance -ClassName Win32_Desktop | Select-Object -ExcludeProperty "Cim*"

SettingID Name                ScreenSaverActive ScreenSaverSecure ScreenSaverTimeout
-----
NT AUTHORITY\SYSTEM False
DEVAPPS\devops34 False
.DEFAULT False

PS C:\Users\devops34>
```

Listing BIOS Information

The WMI **Win32_BIOS** class returns fairly compact and complete information about the system BIOS on the local computer:

PowerShellCopy

```
Get-CimInstance -ClassName Win32_BIOS
```

Listing Processor Information

You can retrieve general processor information by using WMI's **Win32_Processor** class, although you will likely want to filter the information:

PowerShellCopy

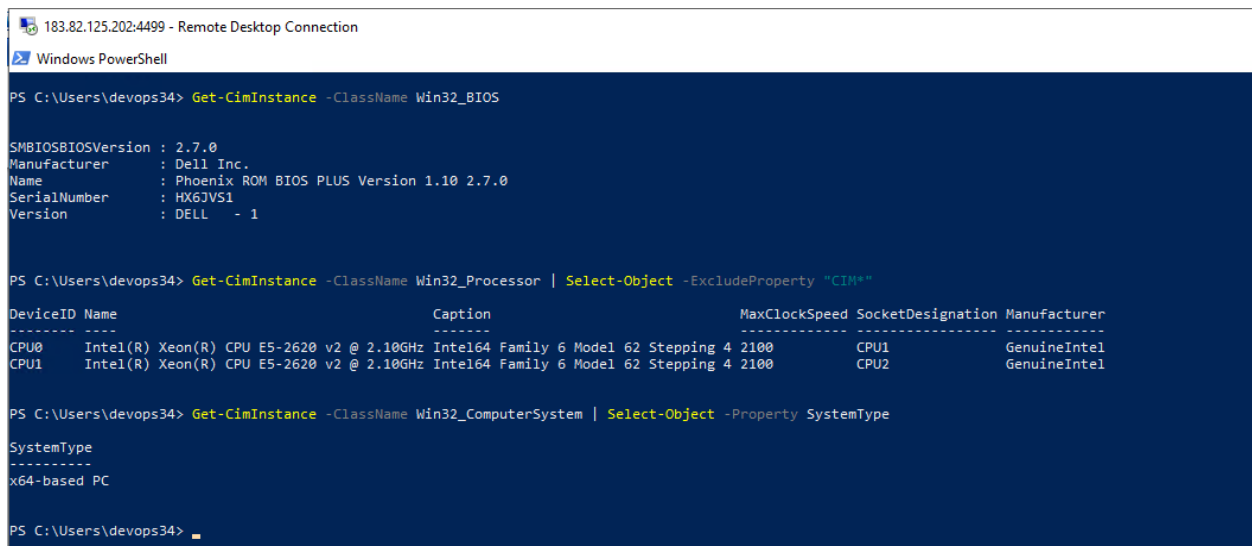
```
Get-CimInstance -ClassName Win32_Processor | Select-Object -ExcludeProperty "CIM*"
```

For a generic description string of the processor family, you can just return the **SystemType** property:

PowerShellCopy

```
Get-CimInstance -ClassName Win32_ComputerSystem | Select-Object -Property SystemType
```

Output :



```
183.82.125.202:4499 - Remote Desktop Connection
Windows PowerShell

PS C:\Users\devops34> Get-CimInstance -ClassName Win32_BIOS

SMBIOSBIOSVersion : 2.7.0
Manufacturer      : Dell Inc.
Name              : Phoenix ROM BIOS PLUS Version 1.10 2.7.0
SerialNumber      : HX6JVS1
Version           : DELL - 1

PS C:\Users\devops34> Get-CimInstance -ClassName Win32_Processor | Select-Object -ExcludeProperty "CIM*"

DeviceID Name                                     Caption                                     MaxClockSpeed SocketDesignation Manufacturer
-----
CPU0      Intel(R) Xeon(R) CPU E5-2620 v2 @ 2.10GHz Intel64 Family 6 Model 62 Stepping 4 2100 CPU1      GenuineIntel
CPU1      Intel(R) Xeon(R) CPU E5-2620 v2 @ 2.10GHz Intel64 Family 6 Model 62 Stepping 4 2100 CPU2      GenuineIntel

PS C:\Users\devops34> Get-CimInstance -ClassName Win32_ComputerSystem | Select-Object -Property SystemType

SystemType
-----
x64-based PC

PS C:\Users\devops34>
```

Listing Computer Manufacturer and Model

Computer model information is also available from **Win32_ComputerSystem**. The standard displayed output will not need any filtering to provide OEM data:

PowerShellCopy

```
Get-CimInstance -ClassName Win32_ComputerSystem
```

Your output from commands such as this, which return information directly from some hardware, is only as good as the data you have. Some information is not correctly configured by hardware manufacturers and may therefore be unavailable.

Listing Installed Hotfixes

You can list all installed hotfixes by using **Win32_QuickFixEngineering**:

PowerShellCopy

```
Get-CimInstance -ClassName Win32_QuickFixEngineering
```

For more succinct output, you may want to exclude some properties. Although you can use the Get-CimInstance's **Property** parameter to choose only the **HotFixID**, doing so will actually return more information, because all the metadata is displayed by default:

PowerShellCopy

```
Get-CimInstance -ClassName Win32_QuickFixEngineering -Property HotFixID
```

OutputCopy

```
183.82.125.202:4499 - Remote Desktop Connection
Windows PowerShell

PS C:\Users\devops34> Get-CimInstance -ClassName Win32_ComputerSystem

Name                PrimaryOwnerName    Domain          TotalPhysicalMemory  Model              Manufacturer
-----                -
DEVAPPS              Windows User         WORKGROUP       103031545856         PowerEdge R720     Dell Inc.

PS C:\Users\devops34> Get-CimInstance -ClassName Win32_QuickFixEngineering

Source      Description      HotFixID      InstalledBy      InstalledOn
-----      -
Update      Update          KB4486153     [redacted]        3/14/2021 12:00:00 AM
Security Update  Security Update  KB5000859     [redacted]        3/14/2021 12:00:00 AM
Security Update  Security Update  KB5000822     [redacted]        3/14/2021 12:00:00 AM

PS C:\Users\devops34> Get-CimInstance -ClassName Win32_QuickFixEngineering -Property HotFixID

InstalledOn      :
Caption          :
Description      :
InstallDate     :
Name            :
Status          :
CSName          :
FixComments     :
HotFixID        : KB4486153
InstalledBy     :
ServicePackInEffect :
PSComputerName  :
CimClass        : root/cimv2:Win32_QuickFixEngineering
CimInstanceProperties : {Caption, Description, InstallDate, Name...}
CimSystemProperties : Microsoft.Management.Infrastructure.CimSystemProperties

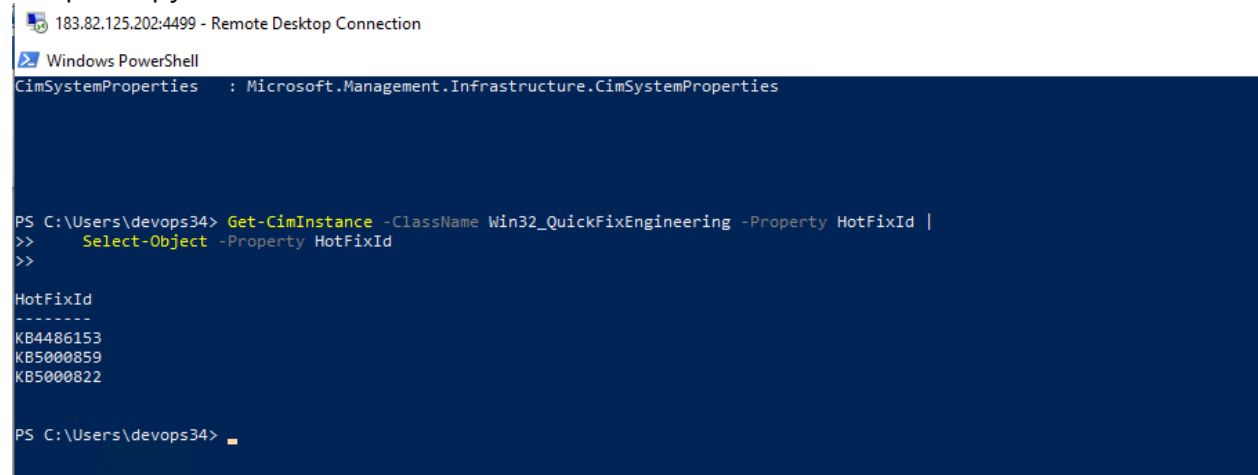
InstalledOn      :
Caption          :
Description      :
InstallDate     :
Name            :
Status          :
```

The additional data is returned, because the **Property** parameter in `Get-CimInstance` restricts the properties returned from WMI class instances, not the object returned to PowerShell. To reduce the output, use `Select-Object`:

PowerShellCopy

```
Get-CimInstance -ClassName Win32_QuickFixEngineering -Property HotFixId |  
    Select-Object -Property HotFixId
```

OutputCopy



The screenshot shows a Windows PowerShell terminal window titled "183.82.125.202:4499 - Remote Desktop Connection". The command prompt shows the execution of the command: `Get-CimInstance -ClassName Win32_QuickFixEngineering -Property HotFixId | Select-Object -Property HotFixId`. The output displays the `HotFixId` property for three instances: `KB4486153`, `KB5000859`, and `KB5000822`.

Listing Operating System Version Information

The **Win32_OperatingSystem** class properties include version and service pack information. You can explicitly select only these properties to get a version information summary from **Win32_OperatingSystem**:

PowerShellCopy

```
Get-CimInstance -ClassName Win32_OperatingSystem |  
    Select-Object -Property  
BuildNumber, BuildType, OSType, ServicePackMajorVersion, ServicePackMinorVersion
```

You can also use wildcards with the `Select-Object`'s **Property** parameter. Because all the properties beginning with either **Build** or **ServicePack** are important to use here, we can shorten this to the following form:

PowerShellCopy

```
Get-CimInstance -ClassName Win32_OperatingSystem | Select-Object -Property  
Build*, OSType, ServicePack*
```

OutputCopy

```
BuildNumber      : 18362  
BuildType        : Multiprocessor Free  
OSType           : 18  
ServicePackMajorVersion : 0  
ServicePackMinorVersion : 0
```

Listing Local Users and Owner

Local general user information — number of licensed users, current number of users, and owner name — can be found with a selection of **Win32_OperatingSystem** class properties. You can explicitly select the properties to display like this:

PowerShellCopy

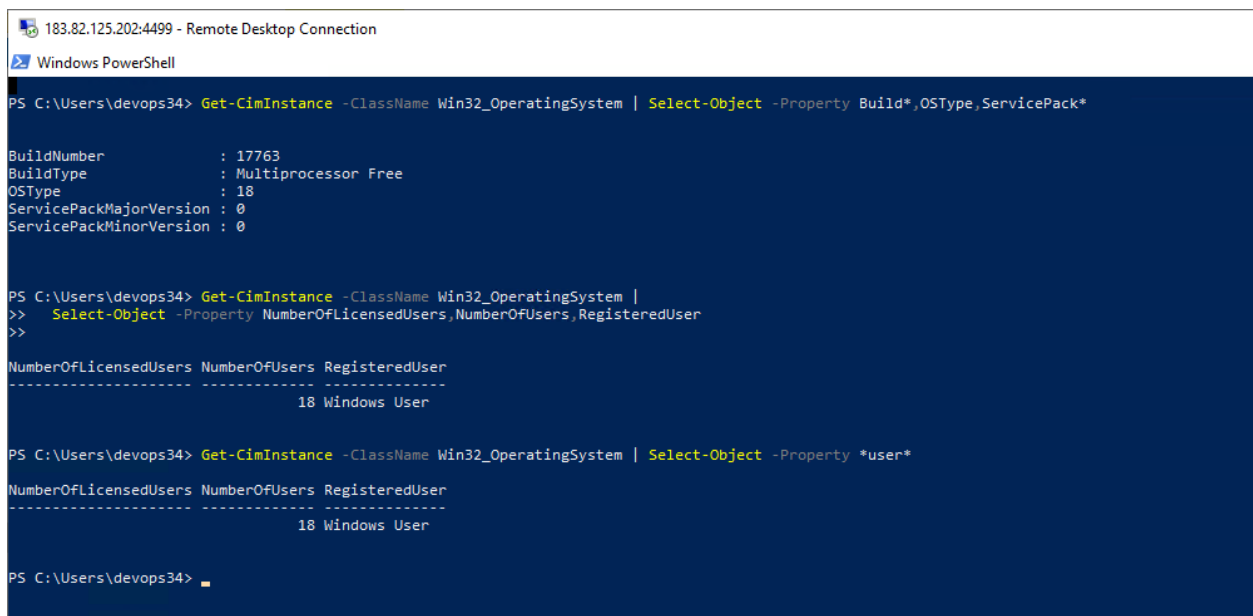
```
Get-CimInstance -ClassName Win32_OperatingSystem |  
    Select-Object -Property NumberOfLicensedUsers,NumberOfUsers,RegisteredUser
```

A more succinct version using wildcards is:

PowerShellCopy

```
Get-CimInstance -ClassName Win32_OperatingSystem | Select-Object -Property *user*
```

Output Copy:



The screenshot shows a Windows PowerShell session titled "183.82.125.202:4499 - Remote Desktop Connection". The user is running three commands to query the Win32_OperatingSystem class. The first command uses Select-Object to filter for Build*, OStype, and ServicePack*. The second command uses Select-Object to filter for NumberOfLicensedUsers, NumberOfUsers, and RegisteredUser. The third command uses Select-Object to filter for *user*. The output of the first command shows system details like BuildNumber (17763), BuildType (Multiprocessor Free), OStype (18), and ServicePack versions. The output of the second and third commands shows a table with three columns: NumberOfLicensedUsers, NumberOfUsers, and RegisteredUser, with a single row of data: 18, Windows, and User.

```
PS C:\Users\devops34> Get-CimInstance -ClassName Win32_OperatingSystem | Select-Object -Property Build*,OStype,ServicePack*  
  
BuildNumber      : 17763  
BuildType        : Multiprocessor Free  
OStype           : 18  
ServicePackMajorVersion : 0  
ServicePackMinorVersion : 0  
  
PS C:\Users\devops34> Get-CimInstance -ClassName Win32_OperatingSystem |  
>>  Select-Object -Property NumberOfLicensedUsers,NumberOfUsers,RegisteredUser  
>>  
  
NumberOfLicensedUsers  NumberOfUsers  RegisteredUser  
-----  
18 Windows User  
  
PS C:\Users\devops34> Get-CimInstance -ClassName Win32_OperatingSystem | Select-Object -Property *user*  
  
NumberOfLicensedUsers  NumberOfUsers  RegisteredUser  
-----  
18 Windows User  
  
PS C:\Users\devops34>
```

Getting Available Disk Space

To see the disk space and free space for local drives, you can use the Win32_LogicalDisk WMI class. You need to see only instances with a DriveType of 3 — the value WMI uses for fixed hard disks.

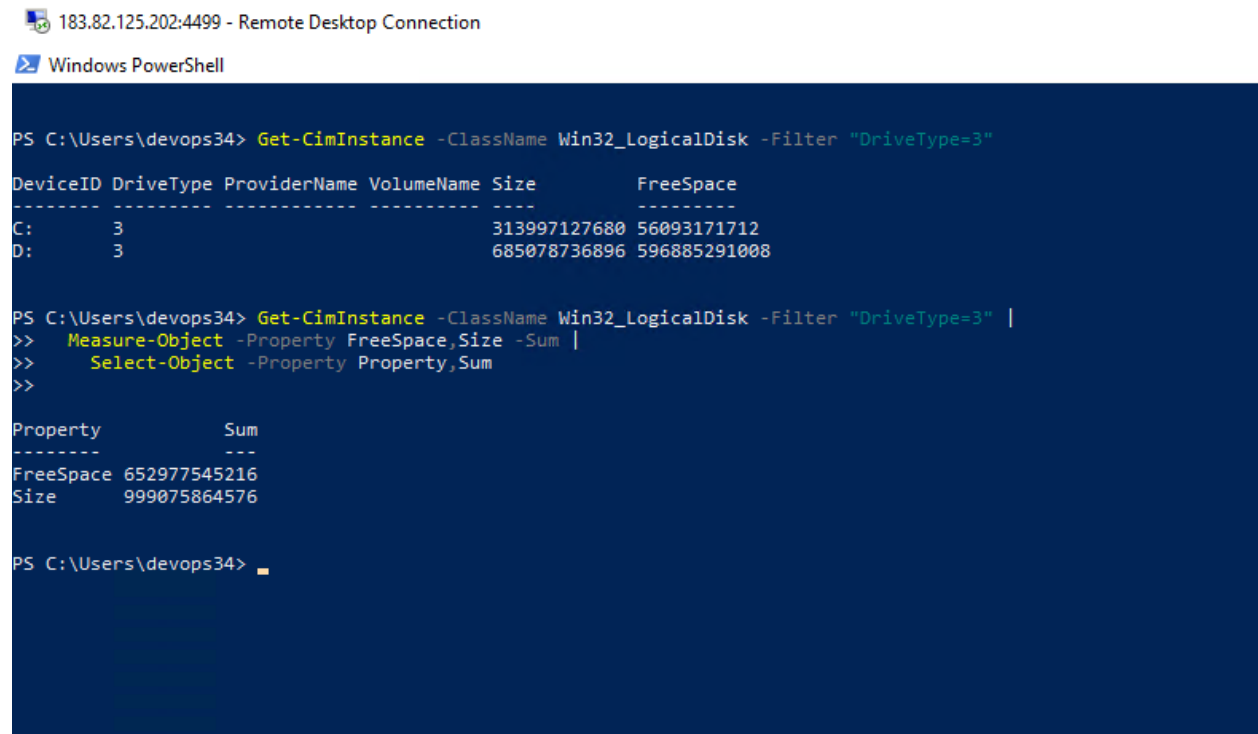
PowerShellCopy

```
Get-CimInstance -ClassName Win32_LogicalDisk -Filter "DriveType=3"
```

PowerShellCopy

```
Get-CimInstance -ClassName Win32_LogicalDisk -Filter "DriveType=3" |  
    Measure-Object -Property FreeSpace,Size -Sum |  
    Select-Object -Property Property,Sum
```

OutputCopy



183.82.125.202:4499 - Remote Desktop Connection

Windows PowerShell

```
PS C:\Users\devops34> Get-CimInstance -ClassName Win32_LogicalDisk -Filter "DriveType=3"  
  
DeviceID DriveType ProviderName VolumeName Size FreeSpace  
-----  
C:        3          C:\          C:         313997127680 56093171712  
D:        3          D:\          D:         685078736896 596885291008  
  
PS C:\Users\devops34> Get-CimInstance -ClassName Win32_LogicalDisk -Filter "DriveType=3" |  
>> Measure-Object -Property FreeSpace,Size -Sum |  
>> Select-Object -Property Property,Sum  
>>  
  
Property      Sum  
-----  
FreeSpace 652977545216  
Size      999075864576  
  
PS C:\Users\devops34> █
```

Getting Logon Session Information

You can get general information about logon sessions associated with users through the **Win32_LogonSession** WMI class:

PowerShellCopy

```
Get-CimInstance -ClassName Win32_LogonSession
```

Getting the User Logged on to a Computer

You can display the user logged on to a particular computer system using Win32_ComputerSystem. This command returns only the user logged on to the system desktop:

PowerShellCopy

```
Get-CimInstance -ClassName Win32_ComputerSystem -Property UserName
```

Output :

```
183.82.125.202:4499 - Remote Desktop Connection
Windows PowerShell

PS C:\Users\devops34> Get-CimInstance -ClassName Win32_LogonSession

LogonId  Name LogonType StartTime          Status AuthenticationPackage
-----
2132282  10      9/6/2022 5:01:28 PM      NTLM
50814414 3      9/7/2022 9:18:57 AM      NTLM

PS C:\Users\devops34> Get-CimInstance -ClassName Win32_ComputerSystem -Property UserName

AdminPasswordStatus      :
BootupState              :
ChassisBootupState       :
KeyboardPasswordStatus    :
PowerOnPasswordStatus     :
PowerSupplyState         :
PowerState               :
FrontPanelResetStatus     :
ThermalState             :
Status                   :
Name                     : DEVAPPS
PowerManagementCapabilities :
PowerManagementSupported  :
Caption                  :
Description              :
InstallDate              :
CreationClassName        :
NameFormat               :
PrimaryOwnerContact       :
PrimaryOwnerName         :
Roles                    :
InitialLoadInfo          :
LastLoadInfo             :
ResetCapability          :
AutomaticManagedPagefile :
AutomaticResetBootOption :
AutomaticResetCapability :
BootOptionOnLimit        :
BootOptionOnWatchDog     :
BootROMSupported         :
BootStatus               :
ChassisSKUNumber         :
CurrentTimeZone          :
```

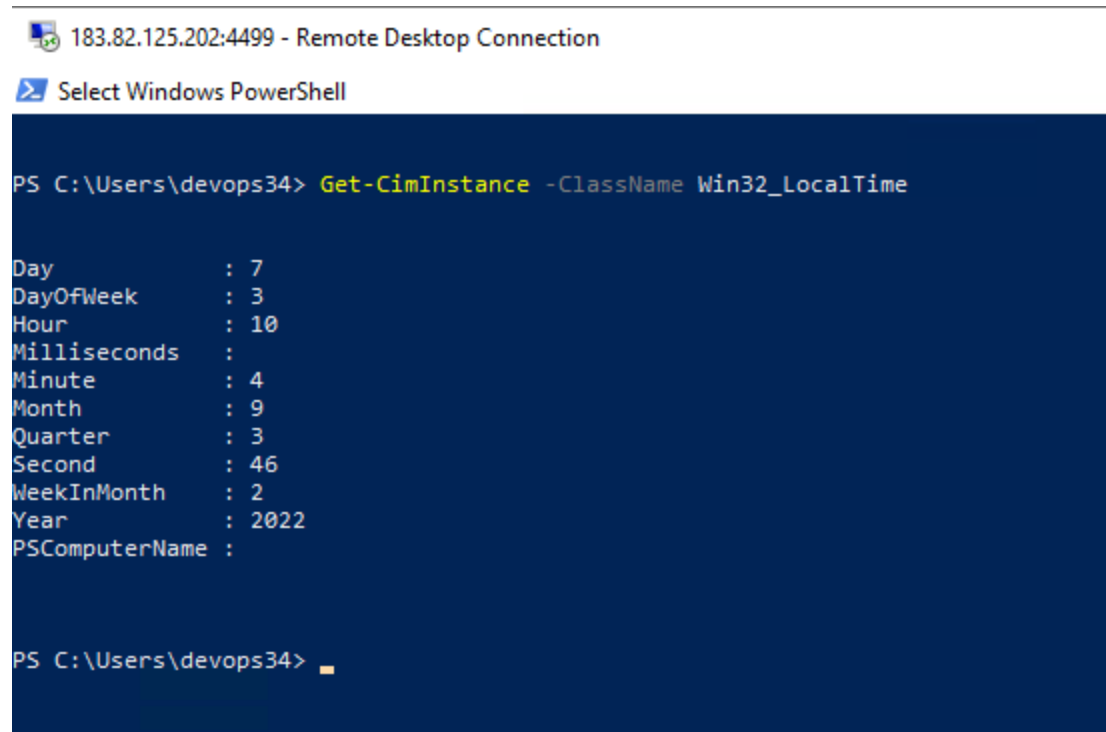
Getting Local Time from a Computer

You can retrieve the current local time on a specific computer by using the **Win32_LocalTime** WMI class.

PowerShellCopy

```
Get-CimInstance -ClassName Win32_LocalTime
```

OutputCopy



183.82.125.202:4499 - Remote Desktop Connection

Select Windows PowerShell

```
PS C:\Users\devops34> Get-CimInstance -ClassName Win32_LocalTime
```

Day	: 7
DayOfWeek	: 3
Hour	: 10
Milliseconds	:
Minute	: 4
Month	: 9
Quarter	: 3
Second	: 46
WeekInMonth	: 2
Year	: 2022
PSComputerName	:

```
PS C:\Users\devops34>
```

Getting Local Time from a Computer

You can retrieve the current local time on a specific computer by using the **Win32_LocalTime** WMI class.

PowerShellCopy

```
Get-CimInstance -ClassName Win32_LocalTime
```

OutputCopy


```
PS C:\Users\devops34> Get-CimInstance -ClassName Win32_Service |
>> Select-Object -Property Status,Name,DisplayName
>>
```

Status	Name	DisplayName
OK	AdobeARMSvc	Adobe Acrobat Update Service
OK	AJRouter	AllJoyn Router Service
OK	ALG	Application Layer Gateway Service
OK	AppHostSvc	Application Host Helper Service
OK	AppIDSvc	Application Identity
OK	AppInfo	Application Information
OK	AppMgmt	Application Management
OK	AppReadiness	App Readiness
OK	AppVClient	Microsoft App-V Client
OK	AppXSvc	AppX Deployment Service (AppXSVC)
OK	aspnet_state	ASP.NET State Service
OK	AudioEndpointBuilder	Windows Audio Endpoint Builder
OK	Audiosrv	Windows Audio
OK	AxInstSV	ActiveX Installer (AxInstSV)
OK	AzureAttestService	AzureAttestService
OK	BFE	Base Filtering Engine
OK	BITS	Background Intelligent Transfer Service
OK	BrokerInfrastructure	Background Tasks Infrastructure Service
OK	BTAGService	Bluetooth Audio Gateway Service
OK	BthAvctpSvc	AVCTP service
OK	bthserv	Bluetooth Support Service
OK	camsvc	Capability Access Manager Service
OK	CDPSvc	Connected Devices Platform Service
OK	CertPropSvc	Certificate Propagation
OK	ClipSVC	Client License Service (ClipSVC)
OK	COMSysApp	COM+ System Application
OK	CoreMessagingRegistrar	CoreMessaging
OK	CryptSvc	Cryptographic Services
OK	CscService	Offline Files
OK	DcomLaunch	DCOM Server Process Launcher
OK	defragsvc	Optimize drives
OK	DeviceAssociationService	Device Association Service
OK	DeviceInstall	Device Install Service
OK	DevQueryBroker	DevQuery Background Discovery Broker
OK	Dhcp	DHCP Client
OK	diagnosticshub.standardcollector.service	Microsoft (R) Diagnostics Hub Standard Collector Service
OK	DiagTrack	Connected User Experiences and Telemetry
OK	DmEnrollmentSvc	Device Management Enrollment Service
OK	dmwappushservice	Device Management Wireless Application Protocol (WAP) Push message Routing Service
OK	Dnscache	DNS Client

Displaying Service Status

To view the status of all services on a specific computer, you can locally use the `Get-Service` cmdlet. For remote systems, you can use the **Win32_Service** WMI class. If you also use `Select-Object` to filter the results to **Status**, **Name**, and **DisplayName**, the output format will be almost identical to that from `Get-Service`:

PowerShellCopy

```
Get-CimInstance -ClassName Win32_Service |
Select-Object -Property Status,Name,DisplayName
```

To allow the complete display of names for the occasional services with extremely long names, you may want to use `Format-Table` with the **AutoSize** and **Wrap** parameters, to optimize column width and allow long names to wrap instead of being truncated:

PowerShellCopy

```
Get-CimInstance -ClassName Win32_Service |
Format-Table -Property Status,Name,DisplayName -AutoSize -Wrap
```

Select Windows PowerShell

```
PS C:\Users\devops34> Get-CimInstance -ClassName Win32_Service |
>> Format-Table -Property Status,Name,DisplayName -AutoSize -Wrap
>>
```

Status	Name	DisplayName
OK	AdobeARMSvc	Adobe Acrobat Update Service
OK	AJRouter	AllJoyn Router Service
OK	ALG	Application Layer Gateway Service
OK	AppHostSvc	Application Host Helper Service
OK	AppIDSvc	Application Identity
OK	AppInfo	Application Information
OK	AppMgmt	Application Management
OK	AppReadiness	App Readiness
OK	AppVClient	Microsoft App-V Client
OK	AppXSvc	AppX Deployment Service (AppXSVC)
OK	aspnet_state	ASP.NET State Service
OK	AudioEndpointBuilder	Windows Audio Endpoint Builder
OK	Audiosrv	Windows Audio
OK	AxInstSV	ActiveX Installer (AxInstSV)
OK	AzureAttestService	AzureAttestService
OK	BFE	Base Filtering Engine
OK	BITS	Background Intelligent Transfer Service
OK	BrokerInfrastructure	Background Tasks Infrastructure Service
OK	BTAGService	Bluetooth Audio Gateway Service
OK	BthAvctpSvc	AVCTP service
OK	bthserv	Bluetooth Support Service
OK	camsvc	Capability Access Manager Service
OK	CDPSvc	Connected Devices Platform Service
OK	CertPropSvc	Certificate Propagation
OK	ClipSVC	Client License Service (ClipSVC)
OK	COMSysApp	COM+ System Application
OK	CoreMessagingRegistrar	CoreMessaging
OK	CryptSvc	Cryptographic Services
OK	CscService	Offline Files
OK	DcomLaunch	DCOM Server Process Launcher
OK	defragsvc	Optimize drives
OK	DeviceAssociationService	Device Association Service
OK	DeviceInstall	Device Install Service
OK	DevQueryBroker	DevQuery Background Discovery Broker
OK	Dhcp	DHCP Client
OK	diagnosticshub.standardcollector.service	Microsoft (R) Diagnostics Hub Standard Collector Service
OK	DiagTrack	Connected User Experiences and Telemetry
OK	DmEnrollmentSvc	Device Management Enrollment Service
OK	dmwappushservice	Device Management Wireless Application Protocol (WAP) Push message Routing Service
OK	Dnscache	DNS Client
OK	DoSvc	Delivery Optimization