WMI Malware: The Complete Forensics Guide

By Alex Detmering

Published: 2025-02-20 · Archived: 2026-04-05 19:13:45 UTC

Attackers can use WMI malware for just about anything. Execution, persistence, lateral movement… honestly, the list goes on. Fortunately for you, there are blogs like these that will help you understand **exactly how bad guys use WMI.**

And **exactly how good guys** – AKA you – **can stop them.**

Now let's get to it.

Jump to…

Introduction to WMI Malware

Windows Management Instrumentation (WMI) is a Microsoft framework for system management that doubles as a versatile tool for bad actors. As DFIR expert Chris Ray explains, "WMI activity is important to review for malicious behavior due to its wide abuse by threat actors. It provides an easy way for threat actors to create processes, tamper with system settings, and perform system recon all without needing to bring in additional tools."

Why it's dangerous:

- Standard Windows feature so threat actors can blend in.
- Enables "fileless" persistence, making file-based scanning useless.
- Allows threat actors to be less reliant on external tools.

How attackers access it:

- PowerShell
- Command prompt
- Various programming interfaces

Use in the attack life cycle:

- Execution
- Discovery
- Defense evasion

- Impact
- Persistence

Now, let's take a closer look at each of these.

Get Technical on WMI

Want to learn more about the technical details of WMI? Here are a few **resources** we recommend:

- **FireEye:** WMI Forensics Whitepaper.
- **0xInfection**: Offensive WMI series.

How Attackers Use WMI Malware

Below are examples of how attackers use WMI malware at each stage of the attack lifecycle.

These examples are useful to investigators **for 2 reasons:**

1. Knowing what to look for and build detection rules off of
2. For running the commands to test out detection rules and look for other artifacts left behind

The examples include multiple ways to do the same thing, so that investigators can make better detection rules that don't just look at WMI, for example.

WMI for Execution

Attackers Can Create a Process Locally
Examples: • Invoke-CimMethod -ClassName win32_process -MethodName create -Arguments @{commandline="notepad.exe"} • Invoke-WmiMethod -Class win32_process -name create -Argumentlist "notepad.exe" • wmic process call create "notepad.exe"
Attackers Can Create a Process Remotely
Examples: • Invoke-CimMethod -ComputerName blocker -ClassName win32_process -MethodName create -Arguments @{commandline="notepad.exe"} • Invoke-WmiMethod -ComputerName blocker -Class win32_process -name create -Argumentlist "notepad.exe" • Wmic /node: blocker process call create "notepad.exe"
Attackers Can Create Services Locally

Examples:

```
Invoke-CimMethod -ClassName Win32_Service -MethodName Create -Arguments @{

     Name             = "Service name"

     DisplayName      = "Service display name"

     PathName         = "%comspec% ping google.com"

     StartMode        = "Automatic"

     StartName        = "LocalSystem"

}
```

- Wmic service call create displayname=”service display name” pathname=”ping google.com” name=”Service name” startmode=”automatic”

Attackers Can Create Services Remotely

Examples:

```
Invoke-CimMethod -ComputerName blocker -ClassName Win32_Service -MethodName Create -Arguments @{

     Name             = "Service name"

     DisplayName      = "Service display name"

     PathName         = "%comspec% ping google.com"

     StartMode        = "Automatic"

     StartName        = "LocalSystem"

}
```

- Wmic /node:blocker service call create displayname=”service display name” pathname=”ping google.com” name=”Service name” startmode=”automatic”

Attackers can use WMI to initiate new processes on both local and remote systems **via the create method for:**

- Win32_Process

- Win32_Service
- Win32_ScheduledJob

The WMIExec.py script is a commonly abused script that uses Win32_Process for remote process creation.

WMI for Discovery

Attackers Can List Directory Content Remotely

Examples:

- Get-CimInstance -ClassName CIM_DataFile -ComputerName blocker -Filter “Drive = ‘C:’ AND Path = ‘\\users\\public\\’”
- Get-WmiObject -Class CIM_DataFile -ComputerName blocker -Filter “Drive = ‘C:’ AND Path = ‘\\users\\public\\’”
- Wmic datafile where “Drive = ‘C:’ AND Path = ‘\\users\\public\\’”

```
PS C:\> Get-CimInstance -ClassName CIM_DataFile -ComputerName blocker -Filter "Drive = 'C:' AND Path = '\\Users\\Public\\'" | Select-Object Name, LastWriteTime, Extension, Size

Name                          LastWriteTime Extension Size
----                          ------------- --------- ----
c:\users\public\desktop.ini                 ini
```

WMI lateral example.

Attackers Can Check Installed Patches

Examples:

- Get-CimInstance -ClassName Win32_QuickFixEngineering
- Get-WmiObject -Class Win32_QuickFixEngineering
- Wmic qfe

```
PS C:\> Get-CimInstance -ClassName Win32_QuickFixEngineering

Source        Description      HotFixID      InstalledBy          InstalledOn
------        -----------      --------      -----------          -----------
              Update           KB5049622     NT AUTHORITY\SYSTEM  2/4/2025 12:00:00 AM
              Security Update  KB5050009     NT AUTHORITY\SYSTEM  2/4/2025 12:00:00 AM
              Security Update  KB5050387     NT AUTHORITY\SYSTEM  2/4/2025 12:00:00 AM
```

WMI_patch level example.

Attackers Can View AV Software

Examples:

- Get-CimInstance -Namespace root/SecurityCenter2 -ClassName AntiVirusProduct
- Get-WmiObject -Namespace root/SecurityCenter2 -Class AntiVirusProduct

```
PS C:\> Get-CimInstance -Namespace root/SecurityCenter2 -ClassName AntiVirusProduct


displayName               : Malwarebytes
instanceGuid              : {0D452135-A081-B000-D6B6-132E52638543}
pathToSignedProductExe    : C:\Program Files\Malwarebytes\Anti-Malware\MBAMWsc.exe
pathToSignedReportingExe  : C:\Program Files\Malwarebytes\Anti-Malware\MBAMWsc.exe
productState              : 397312
timestamp                 : Wed, 05 Feb 2025 02:39:36 GMT
PSComputerName            :

displayName               : Windows Defender
instanceGuid              : {D68DDC3A-831F-4fae-9E44-DA132C1ACF46}
pathToSignedProductExe    : windowsdefender://
pathToSignedReportingExe  : %ProgramFiles%\Windows Defender\MsMpeng.exe
productState              : 393472
timestamp                 : Wed, 05 Feb 2025 02:40:00 GMT
PSComputerName            :
```

WMI AV product example.

Attackers Can View Running Processes

Examples:

- Get-CimInstance -ClassName win32_process -Filter “Name=’lsass.exe'”
- Get-WmiObject -Class win32_process -Filter “Name=’lsass.exe'”
- Wmic process where “name=’lsass.exe'”

```
PS C:\> Get-CimInstance -ClassName win32_process -Filter "Name='lsass.exe'"

ProcessId Name      HandleCount WorkingSetSize VirtualSize
--------- ----      ----------- -------------- -----------
1608      lsass.exe 3030        33337344       2203503026176
```

WMI Process LSASS example.

Attackers Can View Active Services

Examples:

- Get-CimInstance -ClassName Win32_Service -Filter “Name=’windefend'”
- Get-WmiObject -Class win32_Service -Filter “Name=’windefend'”
- Wmic service where “name=’windefend'”

```
PS C:\> Get-CimInstance -ClassName Win32_Service -Filter "Name='windefend'" | fl *


Name                    : WinDefend
Status                  : OK
ExitCode                : 0
DesktopInteract         : False
ErrorControl            : Normal
PathName                : "C:\ProgramData\Microsoft\Windows Defender\Platform\4.18.24090.11-0\MsMpEng.exe"
ServiceType             : Own Process
StartMode               : Manual
Caption                 : Microsoft Defender Antivirus Service
Description             : Helps protect users from malware and other potentially unwanted software
InstallDate             :
CreationClassName       : Win32_Service
Started                 : False
SystemCreationClassName : Win32_ComputerSystem
SystemName              :
AcceptPause             : False
AcceptStop              : False
DisplayName             : Microsoft Defender Antivirus Service
ServiceSpecificExitCode : 0
StartName               : LocalSystem
State                   : Stopped
```

WMI Service Defender example.

Attackers can use WMI to collect a wide range of system data, both locally and remotely.

This includes:

- Checking installed antivirus software via the AntiVirusProduct class.
- Tracking Windows updates and patches via the Win32_QuickFixEngineering class.
- Listing active processes/services via the Win32_Process and Win32_Service classes.
- Listing content of directories via the CIM_DataFile class.

WMI for Defense Evasion

Attackers Can Disable Firewall

Examples:

- Get-CimInstance -Namespace “root/StandardCimv2” -ClassName MSFT_NetFirewallProfile | ForEach-Object { $_.Enabled = $false; $_ | Set-CimInstance }
- Get-WmiObject -Namespace “root\StandardCimv2” -Class MSFT_NetFirewallProfile | ForEach-Object { $_.Enabled = $false; $_.Put() }

Attackers Can Disable Critical Services

Examples:

- Get-CimInstance -ClassName Win32_Service -Filter “Name=’eventlog'” | ForEach-Object { $_.StopService(); $_.ChangeStartMode(‘Disabled’) }
- Get-WmiObject -Class Win32_Service -Filter “Name=’eventlog'” | ForEach-Object { $_.StopService(); $_.ChangeStartMode(‘Disabled’) }
- Wmic service where name=eventlog call ChangeStartMode Disabled

Attackers Can Clear Event Logs

Examples:

- Get-CimInstance -ClassName Win32_NTEventlogFile -Filter “LogFileName=’Application'” | Invoke-CimMethod -MethodName ClearEventlog
- Get-WmiObject -Class Win32_NTEventLogFile -Filter “LogFileName=’Application'”| ForEach-Object { $_.ClearEventLog() }
- Wmic nteventlog where “LogfileName=’Application'” call ClearEventLog

Attackers Can Avoid Sandbox Analysis

Examples:

- Get-CimInstance -ClassName Win32_ComputerSystem | fl *
- Get-WmiObject -Class win32_computersystem | fl *
- Wmic computersystem

```
PS C:\Windows\system32> get-ciminstance -ClassName win32_computersystem |fl hy*, oem*, model, man*


HypervisorPresent : True
OEMLogoBitmap     :
OEMStringArray    : {[MS_VM_CERT/SHA1/27d66596a61c48dd3dc7216fd715126e33f59ae7], Welcome to the Virtual Machine}
model             : VMware7,1
Manufacturer      : VMware, Inc.
```

WMI virtual example.

Attackers can use WMI to tamper with system logs and critical services and evade detection.

This includes:

- Disable Windows Firewalls via MSFT_NetFirewallProfile.
- Disable critical Windows services via Win32_Service.
- Clearing event logs via NTEventLogFile class.
- Checking for virtualization/sandboxing via Win32_computersystem class.

WMI for Impact

Attackers Can Block System Recovery

Examples:

- Get-CimInstance -ClassName Win32_ShadowCopy | Remove-CimInstance
- Get-WmiObject -Class Win32_ShadowCopy | ForEach-Object { $_.Delete() }
- Wmic shadowcopy delete

Attackers Can Force System Reboot

Examples:

- Get-CimInstance -ClassName Win32_OperatingSystem | Invoke-CimMethod -MethodName reboot
- (Get-WmiObject -Class Win32_OperatingSystem).Win32Shutdown(6)
- Wmic os where Primary='TRUE' call Win32Shutdown 6

Attackers can use WMI to disrupt system recovery and force reboots or shutdowns.

Here's how:

- Win32_ShadowCopy class can be used to delete volume shadow copies, inhibiting system recovery.
- Win32_OperatingSystem class methods can force restarts/shutdown.

These tactics are often used in ransomware attacks to make recovery harder or in stealthy intrusions to cover their tracks.

WMI for Persistence

Attackers Can Create WMI Consumers

For examples and more on WMI Consumers for persistence, read:

- How to Investigate Malware WMI Event Consumers 2025.
- How to Find WMI Consumers: Complete Forensic Analysis Guide.

Attackers Can Create Services

See the execution section for examples.

Attackers Can Create Autorun Keys

WMI persistence with autorun example.

Attackers can use WMI to set up persistence mechanisms.

This includes:

- Creating persistent consumers (similar to scheduled tasks).
- Creating services via the Win32_Service Create method.
- Changing registry autorun keys via stdRegProv methods.

A tool called WMIPersis.py simplifies this process by automating event consumer-based persistence.

Real-World Attacks

ShrinkLocker Ransomware

Where WMI Was Used	
Win32_ComputerSystem	• Used to check the computer's domain so it can bail out if the computer is not the intended target. • Perform a system restart when needed.
Win32_OperatingSystem	• Check the OS version. If running on an older system (Windows XP and 2000), it deletes itself and exits.
Win32_OptionalFeature	• Used to check if BitLocker is installed.
StdRegProv	• Used to create registry keys (related to BitLocker).
Win32_PerfRawData_Tcpip_NetworkInterface	• Data returned was used to help generate a random BitLocker password for encryption.
Win32_Service	• Checks to see if BitLocker service is running. If not, then WMI was used to start the service before the encryption takes place.
Win32_Volume	• Used to determine the drive letter for the drive the OS is installed on.
Win32_EncryptableVolume	• Used to check if the disk has been encrypted.

Keep reading about this case

Blue Mockingbird Cryptominer

Where WMI Was Used	
Create persistence via COR_PROFILER	• Used WMIC ENVIRONMENT to create a new system-wide environment variable.

Keep reading about this case

Metador

Where WMI Was Used	
WMI persistence via WMI persistent consumers	• Used the CommandLineEventConsumer class to execute a lolbin cbd.exe 5-6 minutes after the system boots up.

Keep reading about this case

4 WMI Malware Detection Techniques

#1 Monitor for WMI Consumer Persistence	
Why	• Threat actors use WMI consumers for persistence.
Where	• WMI Database • EDR Telemetry • Sysmon logs (events 19, 20, 21) • Microsoft-Windows-WMI-Activity/Operational log (event 4861)
What to look for	• __FilterToConsumerBinding, __EventFilter, __EventConsumer instances outside of the default root\subscription namespace. • Any consumer class not one of the 5 standard consumers. • Instances of ActiveScriptEventConsumer and CommandLineEventConsumer that launch suspicious scripts/processes.
#2 Monitor for Unusual Children of Scrcons.exe	

Why	• Scrcons.exe is the exe responsible for implementing actions defined by ActiveScriptEventConsumer instances. As a result, reviewing the children of scrcons.exe can help find malicious activity executed by WMI persistence.
Where	• Anywhere you get process execution history. Some examples are: ◦ Security log (event 4688) ◦ Sysmon logs (event 1) ◦ EDR Telemetry
What to look for	• Any unusual child processes of scrcons.exe such as powershell.exe, pwsh.exe, cmd.exe, dllhost.exe, etc… Sigma rules already exist to capture some of this activity, such as this rule.
#3 Monitor for Unusual Children of Wmiprvse.exe	
Why	• When WMI is used for remote execution the processes will be children of WmiPrvse.exe.
Where	• Anywhere you get process execution history. Some examples are: ◦ Security log (event 4688) ◦ Sysmon logs (event 1) ◦ EDR Telemetry
What to look for	• Any unusual child processes such as: powershell.exe, cmd.exe, pwsh.exe, reg.exe, etc… Existing rules can be used as a starting point such as this rule.
#4 Monitor Process History for Unusual Usage of WMI Commands	
Why	• Reviewing process command line history allows defenders to find malicious uses of WMI that have been initiated from wmic.exe or powershells.
Where	• Anywhere you get process execution history. Some examples are: ◦ Security log (event 4688) ◦ Sysmon logs (event 1) ◦ EDR Telemetry

What to look for	• Process of wmic.exe, cmd.exe, powershell.exe, pwsh.exe
	• Commandline containing interesting WMI methods like the ones we previously discussed.
	  ◦ Ex. win32_process and call (create a process)
	  ◦ Ex. Win32_ShadowCopy and delete (delete a shadowcopy)
	  ◦ Ex. Win32_NTEventlogFile and ClearEventlog (clear windows event log)

Find WMI Malware Easily

It's important for investigators to understand the fundamentals of WMI, the technical details and classic strategies aren't really required anymore. Once you understand the basics, investigators should focus on the big picture of an investigation.

And it's the job of software to take care of the rest.

Cyber Triage is just such software. Cyber Triage automates the collection of all the artifacts (like WMI) investigators need and scores them according to how suspicious they are using Automated Analysis.

It radically speeds up (and improves the accuracy) of investigations.

Try it today!

Source: https://www.cybertriage.com/blog/wmi-malware/