Mute Sysmon - Silence Sysmon via event manifest tampering

APRIL 23, 2020

GitHub repository: https://github.com/SecurityJosh/MuteSysmon

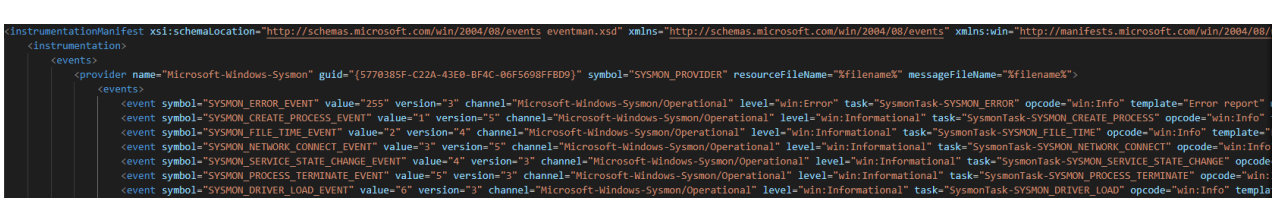
Update 14/07/2020

- To clarify, this technique requires administrative permsissions.
- Twitter user Ring3 API has kindly provided a more precise Sigma rule for the PowerShell activity detection, which can be found here.

Background

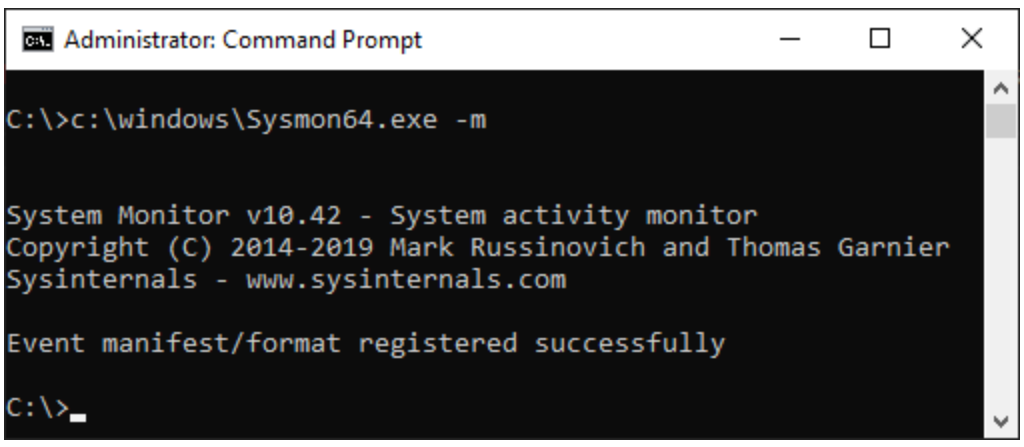
Sysmon is a free tool from Microsoft Sysinternals which logs system activity to the Windows Event Log. It is configurable via rulesets which are written in XML. When combined with a SIEM, it can be used as a flexible endpoint detection system to discover malicious or otherwise anomalous behaviour on a system.

In order to write logs to the Windows Event Log, Sysmon installs an event instrumentation manifest. This manifest describes each type of Sysmon event, including its name and the fields that it contains.

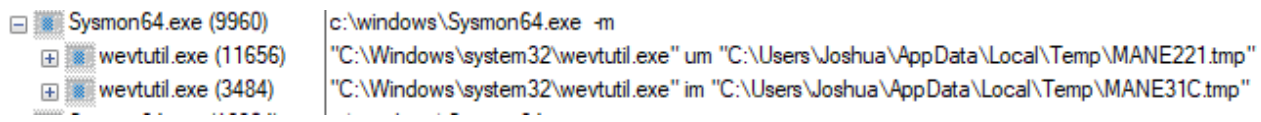


An extract of Sysmon's instrumentation manifest

This manifest is automatically installed when Sysmon is installed, but it can also be installed by passing the `-m` flag to the Sysmon binary:



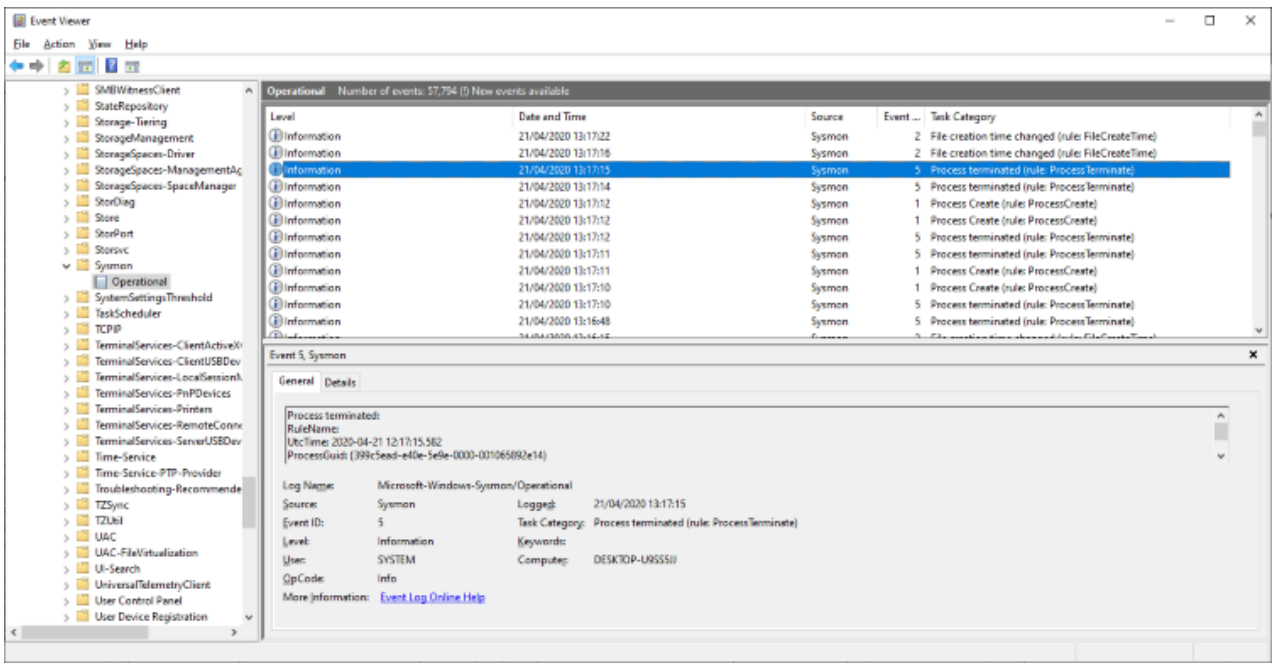
To install the manifest, Sysmon calls the `wevtutil.exe` executable twice. Firstly, it uninstalls any existing Sysmon manifest using the `um` flag. Then, it performs a clean install of the manifest by using the `im` flag:



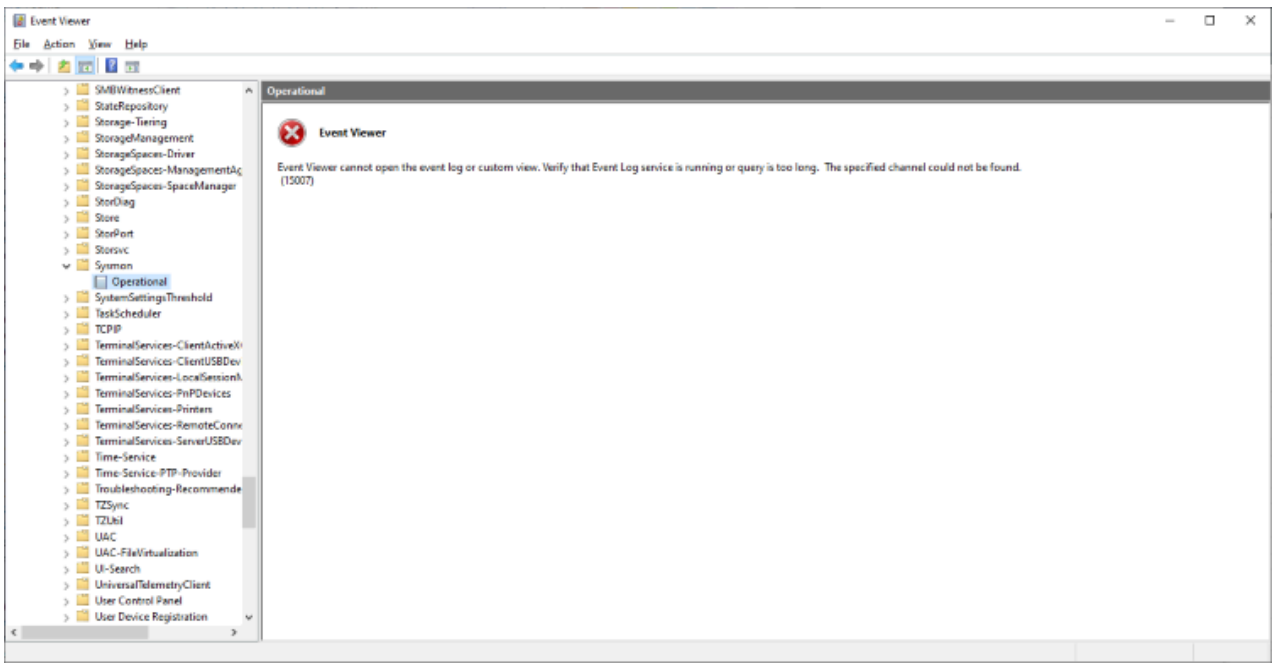
N.B. Although two temporary manifest files are created, the contents of both files is identical.

Event Manifest Tampering

So, if Sysmon installs this manifest as part of its setup process, what happens to its event logging capabilities if we uninstall it?



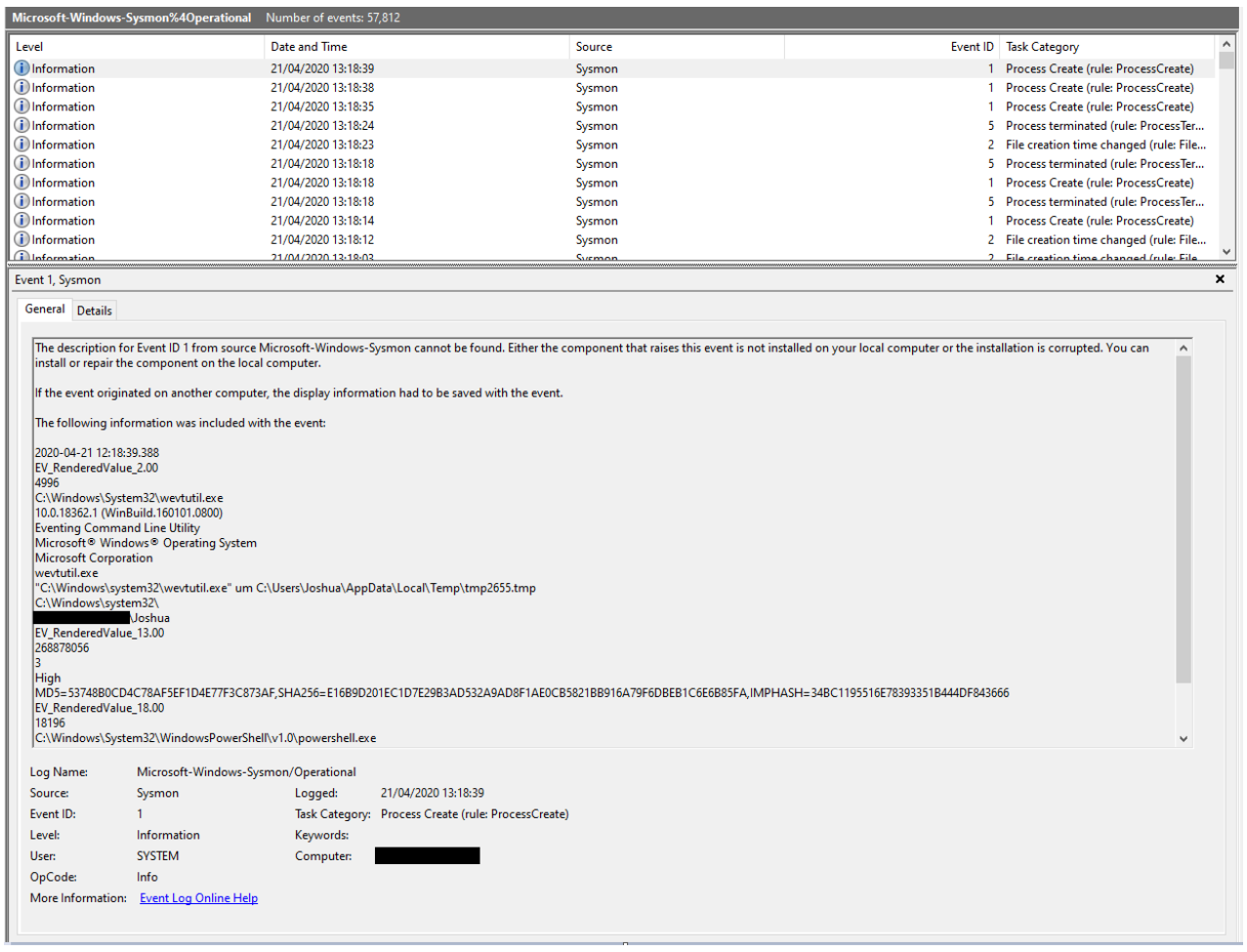
Before manifest uninstallation



After manifest uninstallation

I have written a PowerShell script which automates this uninstallation process, which is available here. Note that uninstalling a manifest only requires a small subset of the entire manifest.

As you can see, the Sysmon channel has been removed. If we open the Sysmon event log file (Located at `C:\Windows\System32\winevt\Logs\Microsoft-Windows-Sysmon%4Operational.evtx`), we can see that the last event that Sysmon wrote was the execution of wevtutil.exe by our PowerShell script. No further system activity is recorded. Sysmon does not manage to write an error event to the Sysmon log.



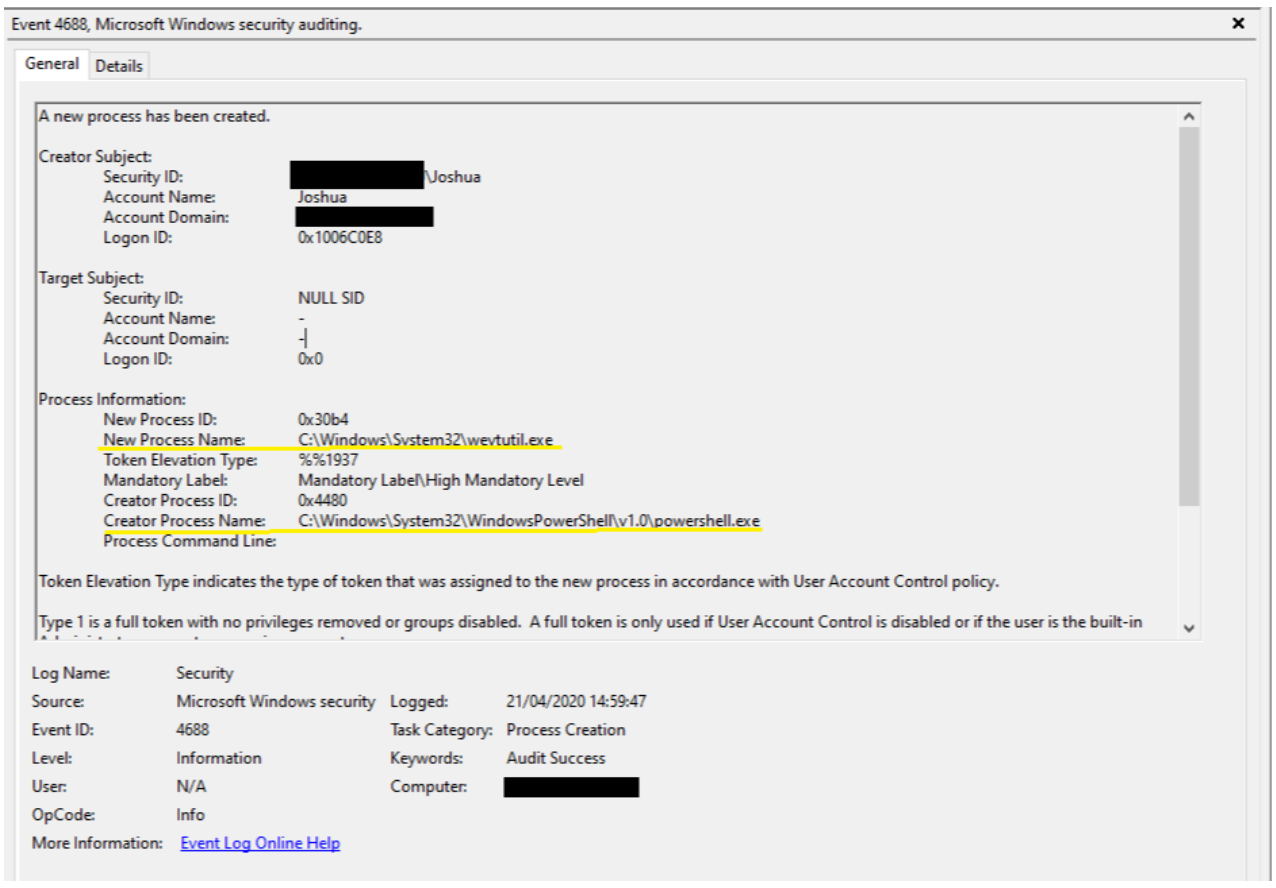
Detection

Depending on how the Sysmon events are shipped off to a SIEM, uninstalling the event manifest may prevent the PowerShell execution event from making it to the SIEM. For instance, a client configured to send logs via the Windows Event Forwarding framework won't be able to find the `Microsoft-Windows-Sysmon/Operational` channel, and so the event won't be forwarded.

As far as I can tell, the event manifest uninstallation is not recorded under any other event channel either.

Using Native Process Creation Events (4688)

It is possible to detect the use of PowerShell to launch wevtutil, which should be considered a suspicious activity.



The following XPath query looks for `C:\Windows\System32\wevtutil.exe` being executed by `C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe`:

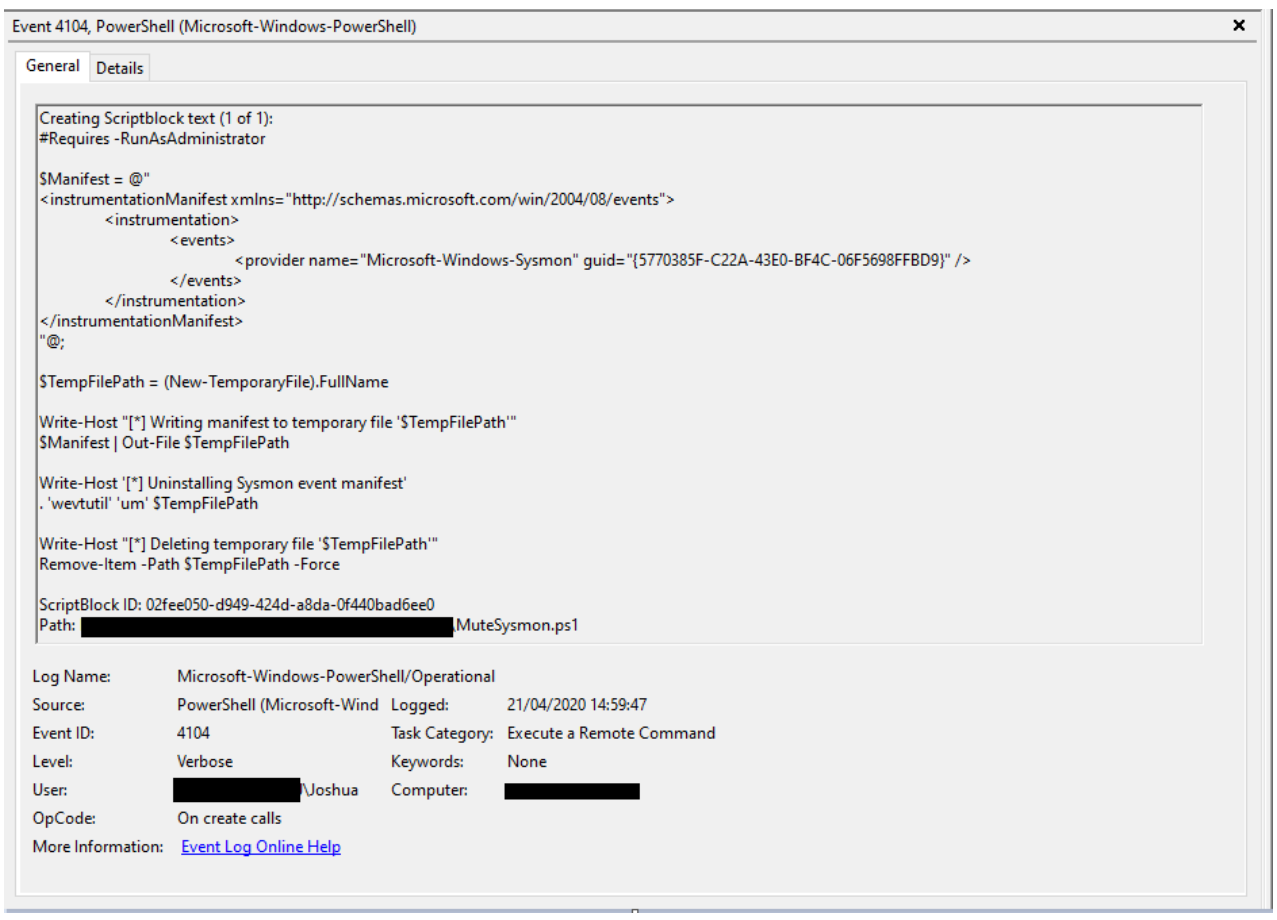
```
<QueryList>
  <Query Id="0" Path="Security">
    <Select Path="Security">*[EventData[Data[@Name='NewProcessNa
  </Query>
</QueryList>
```

This Sigma rule looks for the same activity:

```
title: Potential Event Manifest Tampering
status: experimental
description: Detects potential event manifest tampering
author: SecurityJosh
references:
- https://securityjosh.github.io/Mute_Sysmon.html
tags:
- attack.execution
- attack.t1086
logsource:
  product: windows
  service: security
detection:
  selection:
    EventID: 4688
    "Creator Process Name":
      '*\powershell.exe'
    "New Process Name":
      '*\wevtutil.exe'
  condition: selection
fields:
- "Creator Process Name"
- "New Process Name"
level: medium
falsepositives:
- Unknown
```

Using PowerShell Events (4104)

Although not fool proof, if you are ingesting PowerShell logs into your SIEM, you could look for PowerShell 4104 events which contain the GUID of the Sysmon event log provider:



Sigma Rule:

```
title: PowerShell Execution (Potential event manifest tampering)
description: Detects PowerShell execution with references to the
references:
- https://securityjosh.github.io/Mute_Sysmon.html
author: SecurityJosh
status: experimental
```

```
date: 2020/04/21
logsource:
  product: windows
  service: powershell
  description: Script block logging must be enabled
detection:
  selection:
    EventID: 4104
  keywords:
  - '{5770385F-C22A-43E0-BF4C-06F5698FFBD9}'
  - 'Microsoft-Windows-Sysmon'
  - 'wevtutil'
  condition: selection and all of keywords
level: medium
falsepositives:
- Unknown
tags:
- attack.execution
```

MalwLess configuration

A MalwLess configuration file for the PowerShell configuration file is available here.

Prevention

When uninstalling a manifest using wevtutil.exe, it first attempts to open the corresponding registry key, in this case located at `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\WINEVT\Publishers\{5770385f-c22a-43e0-bf4c-06f5698ffbd9}`. It first requests read access, before requesting full access rights against the registry key.

Once it has that access, it performs the uninstallation routine, which involves deleting a few different registry keys.

Process Name	Operation	Path	Detail	Result
wevtutil.exe	**RegOpenKey**	**HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\WINEVT\Publishers\{5770385f-c22a-43e0-bf4c-06f5698ffbd9}**	**Desired Access: Read**	**SUCCESS**
wevtutil.exe	RegQueryValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\WINEVT\Publishers\{5770385f-c22a-43e0-bf4c-06f5698ffbd9}\(Default)	Type: REG_SZ, Length: 50, Data: Microsoft-Windows-Sysmon	SUCCESS
wevtutil.exe	RegQueryKey	HKLM	Query: HandleTags, HandleTags: 0x0	SUCCESS
wevtutil.exe	**RegOpenKey**	**HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\WINEVT\Publishers\{5770385f-c22a-43e0-bf4c-06f5698ffbd9}**	**Desired Access: All Access**	**SUCCESS**

However, if we prevent wevtutil.exe from acquiring full access rights, the uninstallation quietly aborts, and thus Sysmon continues to function as expected.

I have provided an example PowerShell script, available here, which applies a restricted ACL that prevents the Sysmon event manifest from being uninstalled. It takes a parameter `-IdentityReference` which should be the account used to install Sysmon. This account will be given the full access permissions required to manage Sysmon. The script should be ran after every install / upgrade of Sysmon.

Conclusion

In this blog post, I've explained the purpose of the Sysmon's instrumentation manifest and demonstrated how uninstalling it renders Sysmon unable to log events. I have also covered how this behaviour could potentially be detected and prevented, and provided resources to help with this.

References

- Microsoft Docs - Writing an Instrumentation Manifest
- Uncoder
- MalwLess

#sysmon #redteam #blueteam #powershell #sigma