Sysmon FTW!

BOTCONF 2017 – LIGHTNING TALK

C:\> whoami /all

* Tom Ueltschi
* Swiss Post CERT / SOC / CSIRT, since 2007 (10 years!)
  - Focus: Malware Analysis, Threat Intel, Threat Hunting, Red Teaming
* Talks about «Ponmocup Hunter» (Botconf, DeepSec, SANS DFIR Summit)
* BotConf 2016 talk with same title
* Member of many trust groups / infosec communities
* FIRST SIG member (Malware Analysis, Red Teaming)
* Twitter: @c_APT_ure

BotConf Speaker history

- 2013 - My Name is Hunter, Ponmocup Hunter
- 2014 - Ponmocup Hunter 2.0 – The Sequel
- 2015 - LT: Creating your own CTI (in 3 minutes.. or 5 ☺)
- 2016 - Advanced Incident Detection and Threat Hunting using Sysmon (and Splunk)
- 2017 - CFP got REJECTED (for the first time ☹) … Now what???

Write a book on how to deal with rejection? Maybe not, but…

BotConf Speaker history

- 2013 - My Name is Hunter, Ponmocup Hunter
- 2014 - Ponmocup Hunter 2.0 – The Sequel
- 2015 - LT: Creating your own CTI (in 3 minutes.. or 5 ☺)
- 2016 - Advanced Incident Detection and Threat Hunting using Sysmon (and Splunk)
- 2017 - LT: Sysmon FTW! ☺

or in Botnet terms: turn a "reject" into an "inject"

Secure | https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon

Microsoft Technologies Documentation Resources

Sysinternals Learn Downloads Community

Home / Downloads

Filter

- Learn
- Downloads
  - File and Disk Utilities
  - Networking Utilities
  - Process Utilities
  - Security Utilities
    - Autologon
    - LogonSessions
    - NewSID
    - PsLoggedOn
    - PsLogList
    - RootkitRevealer
    - Sysmon
  - System Information
  - Miscellaneous
  - Sysinternals Suite

Sysmon v6.20

05/22/2017 • 12 minutes to read • Contributors

By Mark Russinovich and Thomas Garnier

Published: November 19, 2017

Download Sysmon (1.4 MB)

Introduction

System Monitor (*Sysmon*) is a Windows system service and device driver that, once installed on a system, remains resident across system reboots to monitor and log system activity to the Windows event log. It provides detailed information about process creations, network connections, and changes to file creation time. By collecting the events it generates using Windows Event Collection or SIEM agents and subsequently analyzing them, you can identify malicious or anomalous activity and understand how intruders and malware operate on your network.

Note that *Sysmon* does not provide analysis of the events it generates, nor does it attempt to protect or hide itself from attackers.

Secure | https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon

Microsoft Technologies Documentation Resources

Sysinternals Learn Downloads Community

Home / Downloads

Filter

- Learn
- Downloads
  - File and Disk Utilities
  - Networking Utilities
  - Process Utilities
  - Security Utilities
    - Autologon
    - LogonSessions
    - NewSID
    - PsLoggedOn
    - PsLogList
    - RootkitRevealer
    - Sysmon
  - System Information
  - Miscellaneous
  - Sysinternals Suite

Sysmon v6.20

05/22/2017 • 12 minutes to read • Contributors

By Mark Russinovich and Thomas Garnier

Published: November 19, 2017

Intr

Systen
remain
provid
time. E
analyz
operat

Note t
itself from attackers.

Event ID 19: WmiEvent (WmiEventFilter activity detected)

When a WMI event filter is registered, which is a method used by malware to execute, this event logs the WMI namespace, filter name and filter expression.

Event ID 20: WmiEvent (WmiEventConsumer activity detected)

This event logs the registration of WMI consumers, recording the consumer name, log, and destination.

Event ID 21: WmiEvent (WmiEventConsumerToFilter activity detected)

When a consumer binds to a filter, this event logs the consumer name and filter path.

Secure | https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon

Microsoft Technologies Documentation Resources

Sysinternals Learn Downloads Community

Sysmon v6.2, AccessChk 6.20, Sigcheck v2.60, Whois v1.20

Rate this article

Mark Russinovich November 22, 2017

Share 0 12 2 0

Sysmon v6.20

This Sysmon release adds the ability to change the Sysmon service and driver names to foil malware that use them to detect its presence.

Sysmon
System Information
Miscellaneous
Sysinternals Suite

analyz
operat

Note t
itself from attackers.

xecute, this event logs the

me, log, and destination.

Event ID 21: WmiEvent (WmiEventConsumerToFilter activity detected)

When a consumer binds to a filter, this event logs the consumer name and filter path.

Secure | https://files.sans.org/summit/Digital_Forensics_and_Incident_Response_Summit_2015/PDFs/TheresSomethingAboutWMIDevonKerr.pdf

MAINTAIN PERSISTENCE

- WMI Persistence requires three components
  - An event filter – the condition we're waiting for
    - _EventFilter objects have a name and a "trigger"
  - An event consumer – the persistence payload
    - _EventConsumer objects have a name and one of the following:
      - A script (contained in objects.data)
      - A path to an external script (somewhere on disk)
      - A path to an executable (not a script, also on disk)
    - Pre-Vista ran as SYSTEM
    - Post-Vista run as LOCAL SERVICE
  - A binding that associates a filter to a consumer
    - _FilterToConsumerBinding objects reference an event filter and an event consumer

MANDIANT A FireEye Company © Mandiant, a FireEye Company. All rights reserved. CONFIDENTIAL

https://www.fireeye.com/content/dam/fireeye-www/global/en/current-threats/pdfs/wp-windows-management-instrumentation.pdf

Windows Management Instrumentation (WMI) Offense, Defense, and Forensics

William Ballenthin, Matt Graeber, Claudiu Teodorescu
FireEye Labs Advanced Reverse Engineering (FLARE) Team,
FireEye, Inc.

FireEye

Malicious WMI Persistence Example

The PowerShell code in Figure 5is a modified instance of the WMI persistence code present in the SEADADDY[13] malware family[14]. The event filter was taken from the PowerSploit persistence module and is designed to trigger shortly after system startup. The event consumer simply executes an executable with SYSTEM privileges.

The event filter in the example in Figure 5 is designed to trigger between 200 and 320 seconds after system startup. Upon triggering the event the event consumer executes an executable that had been previously dropped.
The filter and consumer are registered and bound together by specifying both the filter and consumer within a `__FilterToConsumerBinding` instance.

Figure 5: SEADADDY WMI persistence with PowerShell

```
$filterName='BotFilter82'
$consumerName='BotConsumer23'
$exePath='C:\Windows\System32\evil.exe'
$Query="SELECT * FROM __InstanceModificationEvent
WITHIN 60 WHERE TargetInstance ISA 'Win32_
PerfFormattedData_PerfOS_System' AND
TargetInstance.SystemUpTime >= 200 AND
TargetInstance.SystemUpTime < 320"
$WMIEventFilter=Set-WmiInstance-Class__EventFilter-
NameSpace"root\subscription"-Arguments @
{Name=$filterName;EventNameSpace="root\
cimv2";QueryLanguage="WQL";Query=$Query}
-ErrorActionStop
$WMIEventConsumer=Set-WmiInstance-
ClassCommandLineEventConsumer-Namespace"root\
subscription"-Arguments@=$consumerName;ExecutablePa
th=$exePath;CommandLineTemplate=$exePath}
Set-WmiInstance-Class__FilterToConsumerBinding-
```

Detecting Internal Recon

* 3 or more (of 7) different commands executed within 15 min

```
index=[redacted] sourcetype="WinEventLog:Microsoft-Windows-Sysmon/Operational" ProcessCreate
    (ipconfig OR net.exe OR whoami OR netstat OR nbtstat OR hostname OR tasklist)
```

Whitelisting "known good" processes

```
| search EventCode=1
    Image="*\\ipconfig.exe" OR Image="*\\net.exe" OR Image="*\\whoami.exe" OR Image="*\\netstat.exe" OR
    Image="*\\nbtstat.exe" OR Image="*\\hostname.exe" OR Image="*\\tasklist.exe"
| bin _time span=15m
| rex field=Message ".*User: ([redacted]|NT AUTHORITY)\\\\(?<USER1>.*)"
| stats dc(Image) AS CNT_CMDS values(CommandLine) values(ParentImage) values(ParentCommandLine)
    count by _time ComputerName USER1
| where CNT_CMDS > 2
```

Detecting C2 usingNamed Pipes

* Search for Processes
  - Connecting through Web Proxy and
  - Creating Named Pipes

```
index=[redacted] sourcetype="WinEventLog:Microsoft-Windows-Sysmon/Operational"
    (ProcessCreate OR (NetworkConnect 3128 ([redacted])) OR (PipeEvent "Pipe Created"))
```

Proxy IPs

whitelisting vetted good processes

```
| search EventCode=1 OR EventCode=17 OR
    (EventCode=3 DestinationPort="3128" (DestinationIp="[redacted]"))
| stats dc(TaskCategory) AS Cnt_TaskCat dc(EventCode) AS Cnt_EventCode values(TaskCategory) AS TaskCategory
    values(Image) AS Image values(Hashes) AS Hashes values(PipeName) AS PipeName values(DestinationIp) AS DestinationIp
    count by ComputerName ProcessGuid
| where Cnt_TaskCat >= 2 OR Cnt_EventCode >= 2
| rex field=Hashes ".*MD5=(?<MD5>[A-F0-9]*),IMPHASH=(?<IMPHASH>[A-F0-9]*)"
| stats values(ComputerName) AS Clients values(Image) AS Image values(MD5) AS MD5 values(PipeName) AS PipeName
    count by IMPHASH
| search PipeName="\\*"
```

Proxy IPs

Detecting Mimikatz

* Search for ProcessAccess of LSASS.exe
  - GrantedAccess of: **0x1010, 0x1410, 0x143A**
  - CallTrace: **KERNELBASE.dll and (ntdll.dll or UNKNOWN)**

```
index=[redacted] sourcetype="WinEventLog:Microsoft-Windows-Sysmon/Operational" ProcessAccess lsass.exe
| search TargetImage="*\\lsass.exe"
    ((GrantedAccess="0x1010" OR GrantedAccess="0x1410" OR GrantedAccess="0x143a")
        (CallTrace="*KERNELBASE.dll*" CallTrace="*UNKNOWN*") OR
        (CallTrace="*\\ntdll.dll+4bf9a*" CallTrace="*\\KERNELBASE.dll+189b7*"))
    CallTrace!="*\\fpb.tmp*" CallTrace!="*\\Win64RunProcesses.dll*" CallTrace!="*\\System.ni.dll*" CallTrace!="*\\msi.dll*"
    CallTrace!="*[redacted]
    CallTrace!="*[redacted]
    CallTrace!="*[redacted]
| rex field=CallTrace ".*\\\\ntdll.dll\+(?<NTDLL>[0-9a-fA-F]*)\|.*"
| rex field=CallTrace ".*\\\\KERNELBASE.dll\+(?<KRNLB>[0-9a-fA-F]*)[\|\(].*"
| eval CallTrace2 = replace(CallTrace, "\|", " ") | eval CTLen = len(CallTrace)
| where CTLen > 90
| rename SourceProcessId as srcPID | rename GrantedAccess as GrantAcc
| table _time ComputerName SourceProcessGUID srcPID SourceImage TargetImage GrantAcc NTDLL KRNLB CTLen CallTrace2
| sort _time
```

c-apt-ure.blogspot.fr/2017/12/is-this-blog-still-alive.html

Wednesday, December 6, 2017

Is this blog still alive?

Is this blog still alive? That's a valid question since I haven't blogged for quite some time.
(wow, has it really been more than 3 years!?)
So I finally decided to write another post about some stuff that happened in the meantime...

For the past few years I have been more active on Twitter (@c_APT_ure) and also presenting at conferences and collaborating in closed / trusted groups.

My most recent area of interest has been increasing endpoint visibility using Sysinternals Sysmon and sending logs into Splunk for incident detection and threat hunting.

My first presentation was a year ago at BotConf 2016:

"Advanced Incident Detection and Threat Hunting using Sysmon (and Splunk)"

Slides: https://www.botconf.eu/wp-content/uploads/2016/11/PR12-Sysmon-UELTSCHI.pdf
Video: https://www.youtube.com/watch?v=vv_VXntQTpE

This year I gave an updated version on the same topic at the FIRST annual conference.

Slides: https://www.first.org/resources/papers/conf2017/Advanced-Incident-Detection-and-Threat-Hunting-using-Sysmon-and-Splunk.pdf

There are many good resources for further reading that I can suggest.

- Sysmon - DFIR (Mike Haag / @MHaggis)
  - Sysmon Config files
- ThreatHunter-Playbook (Roberto Rodriguez / @Cyb3rWard0g)
- Operational Look at Sysinternals Sysmon 6.20 Update

The list of resources may get updated every so often...

Thanks for accepting my LT!!

I could not have accepted another rejection

- Twitter: @c_APT_ure
- Blog: http://c-apt-ure.blogspot.com/2017/12/is-this-blog-still-alive.html

→ many resources about Sysmon linked in one place