Use EVTX files on VirusTotal with Timesketch and Sigma (Part 2)

osdfir.blogspot.com/2021/11/use-evtx-files-on-virustotal-part2.html

Alexander Jäger

Use VirusTotal EVTX files to test / verify Sigma rules (Part 2)

This is the second part of a blog series. In the first part we covered manual and automated ways to download a recently added feature of VirusTotal to download EVTX from Sandbox execution. This second part explores ways to use a VirusTotal EVTX file to test a Sigma rule and adjust Sigma config in Timesketch to make the rule work. For this we will use a different sample than in part 1 that matches a rule that would not work out of the box in Timesketch.

Disclaimer

Most of our other blog posts cover open source techniques. The API feature described in this post is part of a commercial offering from VirusTotal and is not available to free tier accounts. Similar files could be created with Cuckoo Sandbox, an open source malware analysis system.

Sigma rule

This article assumes the reader is familiar with basic use of Sigma in Timesketch that was covered in Sigma in Timesketch - let's rule the sketch. To get started we will use a Sigma rule that was developed by Florian Roth and mentioned in a recent talk as the number one among his top 5 Sigma rules. The original rule is available on Github and licensed under the DRL 1.1 license:

Detects a Windows command and scripting interpreter executable started from Microsoft Word, Excel, Powerpoint, Publisher and Visio

Such a behavior is considered bad because it should be very rare that such files execute a command and is often used in phishing and ransomware cases to get the initial foothold.

Now we need a sample, to find a matching one, we go to https://support.virustotal.com/hc/en-us/articles/360017450757-Sigma-Rules-List and select the link in the row for "Microsoft Office Product Spawning Windows Shell".

In the following Search field, we see the search for:

sigma_rule:6a6edfdea6536f74ea66bf73682ed52f4b86435793ed76ff38e3ab0523f029f5

To find even better samples, we will add the following search parameters:

have:evtx p:15+

As we are only interested in samples that have EVTX to be downloaded and samples that are considered bad by more than 15 AV engines.

This reveals among others: adf82ba6ea693dee892b190999cbbe4b6e70ed8d6a9c3e596457de04775bb396. We copy the SHA-256 hash and go back to our shell and run the following command to download the EVTX file and create a new sketch (that was covered in part 1 of the blog series).

dftimewolf vt_evtx_ts adf82ba6ea693dee892b190999cbbe4b6e70ed8d6a9c3e596457de04775bb396 /var/tmp/

After the successful execution, visit the Timesketch instance and the sketch that was created which has around 11k events.

Using an new feature in Timesketch that helps compose new rules, we can copy the rule in a text area:

Translated Sigma rule

The Sigma rule translated by Timesketch creates the following Query (from now on referred as Q1):

(data_type:"windows:evtx:record" AND event_identifier:("1" OR "4688") AND source_name:("Microsoft-Windows-Sysmon" OR "Microsoft-Windows-Security-Auditing" OR "Microsoft-Windows-Eventlog") AND ParentImage:(*WINWORD.EXE OR *EXCEL.EXE OR *POWERPNT.exe OR *MSPUB.exe OR *VISIO.exe OR *OUTLOOK.EXE OR *MSACCESS.EXE OR *EQNEDT32.EXE) AND xml_string:(*cmd.exe OR *powershell.exe

OR *wscript.exe OR *cscript.exe OR *sh.exe OR *bash.exe OR *scrcons.exe OR *schtasks.exe OR *regsvr32.exe OR *hh.exe OR *wmic.exe OR *mshta.exe OR *rundll32.exe OR *msiexec.exe OR *forfiles.exe OR *scriptrunner.exe OR *mftrace.exe OR *AppVLP.exe OR *svchost.exe OR *msbuild.exe))

If we query Timesetch with the translated Sigma rule (Q1) no results (0 events) are returned. However the rule did yield results on VirusTotal.

Let's take a closer look at the Sigma rule and and determine why it is returning no results by querying different parts of the rule independently. These parts are highlighted in bold below:

(data_type:"windows:evtx:record" AND event_identifier:("1" OR "4688") AND source_name:("Microsoft-Windows-Sysmon" OR "Microsoft-Windows-Security-Auditing" OR "Microsoft-Windows-Eventlog") AND ParentImage:(*WINWORD.EXE OR *EXCEL.EXE OR *POWERPNT.exe OR *MSPUB.exe OR *VISIO.exe OR *OUTLOOK.EXE OR *MSACCESS.EXE OR *EQNEDT32.EXE) AND xml_string:(*cmd.exe OR *powershell.exe OR *wscript.exe OR *cscript.exe OR *sh.exe OR *bash.exe OR *scrcons.exe OR *schtasks.exe OR *regsvr32.exe OR *hh.exe OR *wmic.exe OR *mshta.exe OR *rundll32.exe OR *msiexec.exe OR *forfiles.exe OR *scriptrunner.exe OR *mftrace.exe OR *AppVLP.exe OR *svchost.exe OR *msbuild.exe))

We will start with data_type to see all events that have the field data_type set:

data_type:*

That will result in around 11k events. Given the whole sketch has 11 k events it can be assumed all events do have a data_type field, so that is not the problem. The next field to check is:

event_identifier:*

Again 11k events, next in the query is:

source_name:*

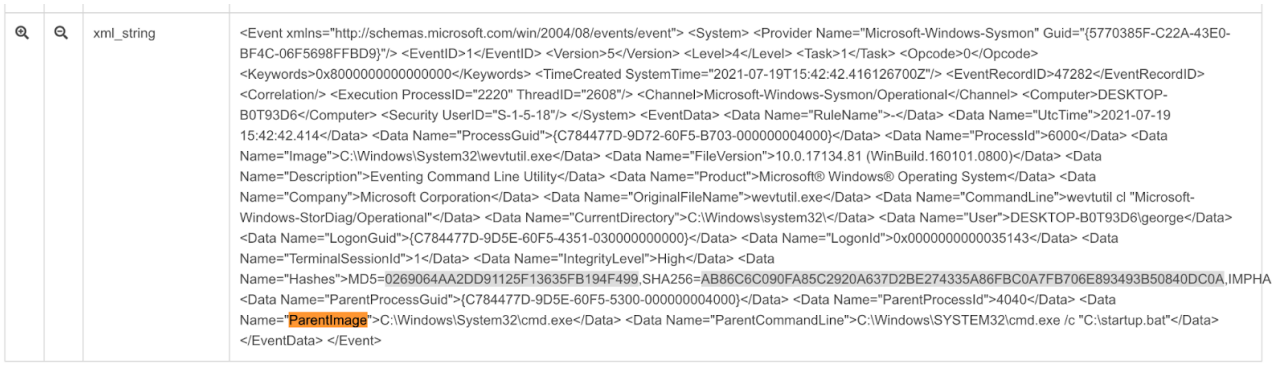
10k events are shown, so not all events have a source_name, but that also does not not seem the problem. Now search for:

ParentImage:*

0 events - now the last one seems problematic. No event has this key as an attribute. But the translated Sigma rule is checking for that field in an AND clause, so if no event has that field set, all the rest of the query could never find any results. Lets search if it exists in any event as part of a value and search for:

ParentImage

The result is 1276 events which have a value that contains ParentImage. That looks much better, now open one event and use the Browser Search function to reveal where the term ParentImage is mentioned - in the xml_string.



		xml_string	<Event xmlns="http://schemas.microsoft.com/win/2004/08/events/event"> <System> <Provider Name="Microsoft-Windows-Sysmon" Guid="{5770385F-C22A-43E0-BF4C-06F5698FFBD9}"/> <EventID>1</EventID> <Version>5</Version> <Level>4</Level> <Task>1</Task> <Opcode>0</Opcode> <Keywords>0x8000000000000000</Keywords> <TimeCreated SystemTime="2021-07-19T15:42:42.416126700Z"/> <EventRecordID>47282</EventRecordID> <Correlation/> <Execution ProcessID="2220" ThreadID="2608"/> <Channel>Microsoft-Windows-Sysmon/Operational</Channel> <Computer>DESKTOP-B0T93D6</Computer> <Security UserID="S-1-5-18"/> </System> <EventData> <Data Name="RuleName">-</Data> <Data Name="UtcTime">2021-07-19 15:42:42.414</Data> <Data Name="ProcessGuid">{C784477D-9D72-60F5-B703-000000004000}</Data> <Data Name="ProcessId">6000</Data> <Data Name="Image">C:\Windows\System32\wevtutil.exe</Data> <Data Name="FileVersion">10.0.17134.81 (WinBuild.160101.0800)</Data> <Data Name="Description">Eventing Command Line Utility</Data> <Data Name="Product">Microsoft® Windows® Operating System</Data> <Data Name="Company">Microsoft Corporation</Data> <Data Name="OriginalFileName">wevtutil.exe</Data> <Data Name="CommandLine">wevtutil cl "Microsoft-Windows-StorDiag/Operational"</Data> <Data Name="CurrentDirectory">C:\Windows\system32\</Data> <Data Name="User">DESKTOP-B0T93D6\george</Data> <Data Name="LogonGuid">{C784477D-9D5E-60F5-4351-030000000000}</Data> <Data Name="LogonId">0x0000000000035143</Data> <Data Name="TerminalSessionId">1</Data> <Data Name="IntegrityLevel">High</Data> <Data Name="Hashes">MD5=0269064AA2DD91125F13635FB194F499,SHA256=AB86C6C090FA85C2920A637D2BE274335A86FBC0A7FB706E893493B50840DC0A,IMPHA <Data Name="ParentProcessGuid">{C784477D-9D5E-60F5-5300-000000004000}</Data> <Data Name="ParentProcessId">4040</Data> <Data Name="ParentImage">C:\Windows\System32\cmd.exe</Data> <Data Name="ParentCommandLine">C:\Windows\SYSTEM32\cmd.exe /c "C:\startup.bat"</Data> </EventData> </Event>

Turns out that a lot of values are stored in this xml_string and that is why the sigma_config.yaml has a lot of field mappings towards the xml_string. So what we would need to do is add the following line to it:

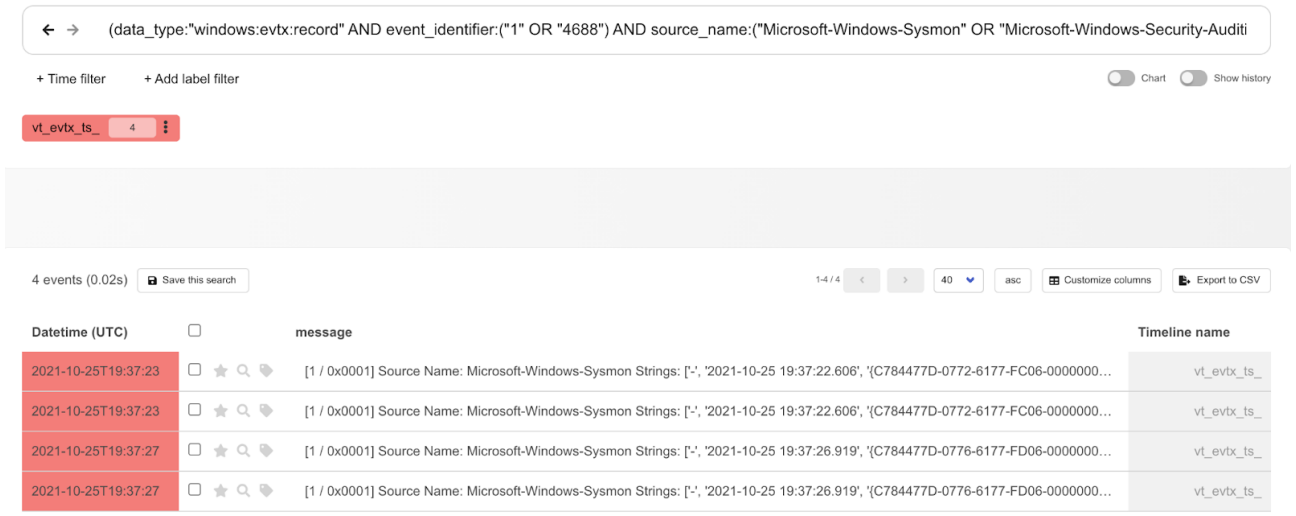
ParentImage: xml_string

Save the file and restart our web server and celery workers. This addition to the sigma_config_yml has already been added to Timesketch with the following Pull Request.

Re-running the compose Sigma rule with the same rule content now gives the following query:

(data_type:"windows:evtx:record" AND event_identifier:("1" OR "4688") AND source_name:("Microsoft-Windows-Sysmon" OR "Microsoft-Windows-Security-Auditing" OR "Microsoft-Windows-Eventlog") AND xml_string:(*WINWORD.EXE OR *EXCEL.EXE OR *POWERPNT.exe OR *MSPUB.exe OR *VISIO.exe OR *OUTLOOK.EXE OR *MSACCESS.EXE OR *EQNEDT32.EXE) AND xml_string:(*cmd.exe OR *powershell.exe OR *wscript.exe OR *cscript.exe OR *sh.exe OR *bash.exe OR *scrcons.exe OR *schtasks.exe OR *regsvr32.exe OR *hh.exe OR *wmic.exe OR *mshta.exe OR *rundll32.exe OR *msiexec.exe OR *forfiles.exe OR *scriptrunner.exe OR *mftrace.exe OR *AppVLP.exe OR *svchost.exe OR *msbuild.exe))

If we query Timesketch with the adjusted Sigma rule, we get 4 results.



And if we look closer into one of the events we can see the following signs of suspicious command (mixed upper- and lowercase, hidden command):

C:\Windows\System32\cmd.exe" /C Powershell.exe -noexit -execUtionPoLiCy ByPAsS -winDoWStYle hidDen -command $nhqId…

We could now store the Sigma rule file in the Timesketch folder according to the documentation and run the analyzer and mark the events and future events in other investigations. As we checked the rule and added mappings, we are also good to add an entry to the central file called sigma_blocklist.csv in the Timesketch project by making a pull request that keeps track of considered good, problematic and unchecked rules.

You can repeat the last steps for other rules with other samples, the Timesketch team highly appreciates contributions for the community to get more rules checked and validated with specific samples so that they can be re-validated by other people.

How can this be used on a larger scale?

Another possibility is to download a larger set of event log files and then push them into our existing SIEM and test our detection and / or log parsing pipeline. The advantage is that the provided data is as diverse in structure, format and content as we could expect from our real endpoints.

Saving and re-using the adjusted Sigma rules (the Timesketch team is working on reducing those needs to adjust) will allow analysts to reproduce investigation steps more quickly and reliably, as they will not have to remember queries but simply run existing rules over the sketch.

Learn more

To learn more about the VirusTotal EVTX information, visit the VirusTotal EVTX API page.

If you have any questions please reach out on the Open Source DFIR Slack community.

Incident Response in the Cloud

Parsing the $MFT NTFS metadata file

Forensic Disk Copies in GCP & AWS