Threat Hunter Playbook

Search this book...

KNOWLEDGE LIBRARY

Windows

PRE-HUNT ACTIVITIES

Data Management

GUIDED HUNTS

Windows

- LSASS Memory Read Access
- DLL Process Injection via CreateRemoteThread and LoadLibrary
- Active Directory Object Access via Replication Services
- Active Directory Root Domain Modification for Replication Services
- Registry Modification to Enable Remote Desktop Conections
- Local PowerShell Execution
- WDigest Downgrade
- PowerShell Remote Session
- Alternate PowerShell Hosts
- Domain DPAPI Backup Key Extraction
- SysKey Registry Keys Access
- SAM Registry Hive Handle Request
- WMI Win32_Process Class and Create Method for Remote Execution
- WMI Eventing
- WMI Module Load
- Local Service Installation
- Remote Service creation
- Remote Service Control Manager Handle
- Remote Interactive Task Manager LSASS Dump
- Registry Modification for Extended

Remote WMI ActiveScriptEventConsumers

Hypothesis

Adversaries might be leveraging WMI ActiveScriptEventConsumers remotely to move laterally in my network.

Technical Context

One of the components of an Event subscription is the event consumer. It is basically the main action that gets executed when a filter triggers (i.e. monitor for authentication events. if one occurs. trigger the consumer).

According to MS Documentation, there are several WMI consumer classes available

- ActiveScriptEventConsumer -> Executes a predefined script in an arbitrary scripting language when an event is delivered to it. Example -> Running a Script Based on an Event
- CommandLineEventConsumer -> Launches an arbitrary process in the local system context when an event is delivered to it. Example -> Running a Program from the Command Line Based on an Event
- LogFileEventConsumer -> Writes customized strings to a text log file when events are delivered to it. Example -> Writing to a Log File Based on an Event
- NTEventLogEventConsumer -> Logs a specific Message to the Windows event log when an event is delivered to it. Example -> Logging to NT Event Log Based on an Event
- ScriptingStandardConsumerSetting Provides registration data common to all instances of the ActiveScriptEventConsumer class.
- SMTPEventConsumer Sends an email Message using SMTP each time an event is delivered to it. Example -> Sending Email Based on an Event

The ActiveScriptEventConsumer class allows for the execution of scripting code from either JScript or VBScript engines. Finally, the WMI script host process is `%SystemRoot%\system32\wbem\scrcons.exe`.

Offensive Tradecraft

Threat actors can achieve remote code execution by using WMI event subscriptions. Normally, a permanent WMI event subscription is designed to persist and respond to certain events. According to Matt Graeber, if an attacker wanted to execute a single payload however, the respective event consumer would just need to delete its corresponding event filter, consumer, and filter to consumer binding. The advantage of this technique is that the payload runs as SYSTEM, and it avoids having a payload be displayed in plaintext in the presence of command line auditing.

Pre-Recorded Security Datasets

Metadata	Value
docs	https://securitydatasets.com/notebooks/atomic/windows/lateral_movement/SDWIN-200724174200.html
link	https://raw.githubusercontent.com/OTRF/Security-Datasets/master/datasets/atomic/windows/lateral_movement/host/covenant_wmi_remote_event_subscription_ActiveS

Download Dataset

```
import requests
from zipfile import ZipFile
from io import BytesIO

url = 'https://raw.githubusercontent.com/OTRF/Security-Datasets/master/datasets
zipFileRequest = requests.get(url)
zipFile = ZipFile(BytesIO(zipFileRequest.content))
datasetJSONPath = zipFile.extract(zipFile.namelist()[0])
```

Read Dataset

```
import pandas as pd
from pandas.io import json

df = json.read_json(path_or_buf=datasetJSONPath, lines=True)
```

Analytics

A few initial ideas to explore your data and validate your detection logic:

Analytic I

Look for the creation of Event consumers of script type.

Data source	Event Provider	Relationship	Event
WMI object	Microsoft-Windows-Sysmon/Operational	User created Wmi consumer	20

Logic

```
SELECT EventID, EventType
FROM dataTable
WHERE Channel = 'Microsoft-Windows-Sysmon/Operational'
  AND EventID = 20
  AND LOWER(Message) Like '%type: script%'
```

Pandas Query

```
(
df[['@timestamp','Hostname','EventID','EventType','Message']]

[(df['Channel'] == 'Microsoft-Windows-Sysmon/Operational')
    & (df['EventID'] == 20)
```

```
    & (df['Message'].str.lower().str.contains('.*type: script.*', regex=True))
]
)
```

Analytic II

Look for the creation of Event consumers of script type (i.e vbscript).

Data source	Event Provider	Relationship	Event
WMI object	Microsoft-Windows-WMI-Activity/Operational	Wmi subscription created	5861

Logic

```
SELECT EventID, SourceName
FROM dataTable
WHERE Channel = 'Microsoft-Windows-WMI-Activity/Operational'
  AND EventID = 5861
  AND LOWER(Message) LIKE '%scriptingengine = "vbscript"%'
```

Pandas Query

```
(
df[['@timestamp','Hostname','EventID','SourceName','Message']]

[(df['Channel'] == 'Microsoft-Windows-WMI-Activity/Operational')
    & (df['EventID'] == 5861)
    & (df['Message'].str.lower().str.contains('.*scriptingengine = "vbscript".*
]
)
```

Analytic III

Look for any indicators that the WMI script host process
`%SystemRoot%\system32\wbem\scrcons.exe` is created. This is created by svchost.exe.

Data source	Event Provider	Relationship	Event
Process	Microsoft-Windows-Sysmon/Operational	Process created Process	1

Logic

```
SELECT ParentImage, Image, CommandLine, ProcessId, ProcessGuid
FROM dataTable
WHERE Channel = "Microsoft-Windows-Sysmon/Operational"
    AND EventID = 1
    AND Image LIKE '%scrcons%'
```

Pandas Query

```
(
df[['@timestamp','Hostname','ParentImage','Image','CommandLine','ProcessId','Pr

[(df['Channel'] == 'Microsoft-Windows-Sysmon/Operational')
    & (df['EventID'] == 1)
```

```
    & (df['Image'].str.lower().str.contains('.*scrcons.*', regex=True))
]
)
```

Analytic IV

Look for any indicators that the WMI script host process `%SystemRoot%\system32\wbem\scrcons.exe` is created. This is created by svchost.exe.

Data source	Event Provider	Relationship	Event
Process	Microsoft-Windows-Security-Auditing	Process created Process	4688

Logic

```
SELECT ParentProcessName, NewProcessName, CommandLine, NewProcessId
FROM dataTable
WHERE LOWER(Channel) = "security"
    AND EventID = 4688
    AND NewProcessName LIKE '%scrcons%'
```

Pandas Query

```
(
df[['@timestamp','Hostname','ParentProcessName','NewProcessName','CommandLine',

[(df['Channel'].str.lower() == 'security')
    & (df['EventID'] == 4688)
    & (df['NewProcessName'].str.lower().str.contains('.*scrcons.*', regex=True)
]
)
```

Analytic V

Look for any indicators that the WMI script host process `%SystemRoot%\system32\wbem\scrcons.exe` is being used. You can do this by looking for a few modules being loaded by a process.

Data source	Event Provider	Relationship	Event
Module	Microsoft-Windows-Sysmon/Operational	Process loaded Dll	7

Logic

```
SELECT Image, ImageLoaded, Description, ProcessGuid
FROM dataTable
WHERE Channel = "Microsoft-Windows-Sysmon/Operational"
    AND EventID = 7
    AND LOWER(ImageLoaded) IN (
        'c:\\windows\\system32\\wbem\\scrcons.exe',
        'c:\\windows\\system32\\vbscript.dll',
        'c:\\windows\\system32\\wbem\\wbemdisp.dll',
        'c:\\windows\\system32\\wshom.ocx',
        'c:\\windows\\system32\\scrrun.dll'
    )
```

Pandas Query

```
(
df[['@timestamp','Hostname','Image','ImageLoaded','Description','ProcessGuid']]

[(df['Channel'] == 'Microsoft-Windows-Sysmon/Operational')
    & (df['EventID'] == 7)
    & (df['ImageLoaded'].str.lower().isin(['c:\\windows\\system32\\wbem\\scrcor
        'c:\\windows\\system32\\\vbscript.dll',
        'c:\\windows\\system32\\wbem\\\wbemdisp.dll',
        'c:\\windows\\system32\\wshom.ocx',
        'c:\\windows\\system32\\scrrun.dll'
    ]))
]
)
```

Analytic VI

Look for any indicators that the WMI script host process `%SystemRoot%\system32\wbem\scrcons.exe` is being used and add some context to it that might not be normal in your environment. You can add network connections context to look for any scrcons.exe reaching out to external hosts over the network.

Data source	Event Provider	Relationship	Event
Process	Microsoft-Windows-Sysmon/Operational	Process created Process	1
Process	Microsoft-Windows-Sysmon/Operational	Process connected to Ip	3
Module	Microsoft-Windows-Sysmon/Operational	Process loaded Dll	7

Logic

```
SELECT d.`@timestamp`, c.Image, d.DestinationIp, d.ProcessId
FROM dataTable d
INNER JOIN (
    SELECT b.ImageLoaded, a.CommandLine, b.ProcessGuid, a.Image
    FROM dataTable b
    INNER JOIN (
        SELECT ProcessGuid, CommandLine, Image
        FROM dataTable
        WHERE Channel = "Microsoft-Windows-Sysmon/Operational"
            AND EventID = 1
            AND Image LIKE '%scrcons.exe'
        ) a
    ON b.ProcessGuid = a.ProcessGuid
    WHERE b.Channel = "Microsoft-Windows-Sysmon/Operational"
        AND b.EventID = 7
        AND LOWER(b.ImageLoaded) IN (
            'c:\\\windows\\\system32\\\wbem\\\scrcons.exe',
            'c:\\\windows\\\system32\\\\vbscript.dll',
            'c:\\\windows\\\system32\\\wbem\\\wbemdisp.dll',
            'c:\\\windows\\\system32\\\wshom.ocx',
            'c:\\\windows\\\system32\\\scrrun.dll'
        )
) c
ON d.ProcessGuid = c.ProcessGuid
WHERE d.Channel = "Microsoft-Windows-Sysmon/Operational"
    AND d.EventID = 3
```

Pandas Query

```
imageLoadDf = (
df[['@timestamp','Hostname','Image','ImageLoaded','Description','ProcessGuid']]

[(df['Channel'] == 'Microsoft-Windows-Sysmon/Operational')
    & (df['EventID'] == 7)
    & (df['ImageLoaded'].str.lower().isin(['c:\\windows\\system32\\wbem\\scrcor
        'c:\\windows\\system32\\\vbscript.dll',
        'c:\\windows\\system32\\wbem\\\wbemdisp.dll',
        'c:\\windows\\system32\\wshom.ocx',
        'c:\\windows\\system32\\scrrun.dll'
    ]))
]
)

processCreateDf = (
df[['@timestamp','Hostname','ParentImage','Image','CommandLine','ProcessId','Pr

[(df['Channel'] == 'Microsoft-Windows-Sysmon/Operational')
    & (df['EventID'] == 1)
    & (df['Image'].str.lower().str.contains('.*scrcons.*', regex=True))
]
)

firstJoinDf = (
pd.merge(imageLoadDf, processCreateDf,
    on = 'ProcessGuid', how = 'inner')
)

networkConnectionDf = (
df[['@timestamp','Hostname','Image','DestinationIp','ProcessGuid']]

[(df['Channel'] == 'Microsoft-Windows-Sysmon/Operational')
    & (df['EventID'] == 3)
]
)

(
pd.merge(firstJoinDf, networkConnectionDf,
    on = 'ProcessGuid', how = 'inner')
)
```

Analytic VII

One of the main goals is to find context that could tell us that `scrcons.exe` was used over the network (Lateral Movement). One way would be to add a network logon session as context to some of the previous events.

Data source	Event Provider	Relationship	Event
Process	Microsoft-Windows-Sysmon/Operational	Process created Process	1
Module	Microsoft-Windows-Sysmon/Operational	Process loaded Dll	7
Authentication log	Microsoft-Windows-Security-Auditing	User authenticated Host	4624

Logic

```
SELECT d.`@timestamp`, d.TargetUserName, c.Image, c.ProcessId
FROM dataTable d
INNER JOIN (
```

```
    SELECT b.ImageLoaded, a.CommandLine, b.ProcessGuid, a.Image, b.ProcessId
    FROM dataTable b
    INNER JOIN (
        SELECT ProcessGuid, CommandLine, Image
        FROM dataTable
        WHERE Channel = "Microsoft-Windows-Sysmon/Operational"
            AND EventID = 1
            AND Image LIKE '%scrcons.exe'
        ) a
    ON b.ProcessGuid = a.ProcessGuid
    WHERE b.Channel = "Microsoft-Windows-Sysmon/Operational"
        AND b.EventID = 7
        AND LOWER(b.ImageLoaded) IN (
            'c:\\\windows\\\system32\\\wbem\\\scrcons.exe',
            'c:\\\windows\\\system32\\\\vbscript.dll',
            'c:\\\windows\\\system32\\\wbem\\\wbemdisp.dll',
            'c:\\\windows\\\system32\\\wshom.ocx',
            'c:\\\windows\\\system32\\\scrrun.dll'
        )
) c
ON split(d.ProcessId, '0x')[1] = LOWER(hex(CAST(c.ProcessId as INT)))
WHERE LOWER(d.Channel) = "security"
    AND d.EventID = 4624
    AND d.LogonType = 3
```

Pandas Query

```
processCreateDf = (
df[['@timestamp','Hostname','ParentImage','Image','CommandLine','ProcessId','Pr

[(df['Channel'] == 'Microsoft-Windows-Sysmon/Operational')
    & (df['EventID'] == 1)
    & (df['Image'].str.lower().str.contains('.*scrcons.*', regex=True))
]
)

imageLoadDf = (
df[['@timestamp','Hostname','Image','ImageLoaded','Description','ProcessGuid']]

[(df['Channel'] == 'Microsoft-Windows-Sysmon/Operational')
    & (df['EventID'] == 7)
    & (df['ImageLoaded'].str.lower().isin(['c:\\windows\\system32\\wbem\\scrcor
        'c:\\windows\\system32\\\vbscript.dll',
        'c:\\windows\\system32\\wbem\\\wbemdisp.dll',
        'c:\\windows\\system32\\wshom.ocx',
        'c:\\windows\\system32\\scrrun.dll'
    ]))
]
)

firstJoinDf = (
pd.merge(processCreateDf, imageLoadDf,
    on = 'ProcessGuid', how = 'inner')
)

firstJoinDf['ProcessId'] = firstJoinDf['ProcessId'].apply(int).apply( hex )

networkLogonDf = (
df[['@timestamp','Hostname','TargetUserName','TargetLogonId','IpAddress','Proce

[(df['Channel'].str.lower() == 'security')
    & (df['EventID'] == 4624)
    & (df['LogonType'] == 3)
]
)

(
pd.merge(firstJoinDf, networkLogonDf,
    on = 'ProcessId', how = 'inner')
)
```

Analytic VIII

One of the main goals is to find context that could tell us that scrcons.exe was used over the network (Lateral Movement). One way would be to add a network logon session as context to some of the previous events.

Data source	Event Provider	Relationship	Event
Authentication log	Microsoft-Windows-Security-Auditing	User authenticated Host	4624

Logic

```
SELECT `@timestamp`, TargetUserName,ImpersonationLevel, LogonType, ProcessName
FROM dataTable
WHERE LOWER(Channel) = "security"
    AND EventID = 4624
    AND LogonType = 3
    AND ProcessName LIKE '%scrcons.exe'
```

Pandas Query

```
(
df[['@timestamp','Hostname','TargetUserName','ImpersonationLevel','LogonType','

[(df['Channel'].str.lower() == 'security')
    & (df['EventID'] == 4624)
    & (df['LogonType'] == 3)
    & (df['ProcessName'].str.lower().str.endswith('scrcons.exe', na=False))
]
)
```

Known Bypasses

False Positives

Hunter Notes

- Baseline your environment to identify normal activity. Apparently, SCCM leverages WMI event subscriptions.

Hunt Output

Type	Link
Sigma Rule	https://github.com/SigmaHQ/sigma/blob/master/rules/windows/builtin/security/win_scrcons_remote_wmi_scripteventcons

References

- https://www.mdsec.co.uk/2020/09/i-like-to-move-it-windows-lateral-movement-part-1-wmi-event-subscription/

- https://www.fireeye.com/content/dam/fireeye-www/services/pdfs/sans-dfir-2015.pdf
- https://www.blackhat.com/docs/us-15/materials/us-15-Graeber-Abusing-Windows-Management-Instrumentation-WMI-To-Build-A-Persistent Asynchronous-And-Fileless-Backdoor-wp.pdf
- https://docs.microsoft.com/en-us/windows/win32/wmisdk/scriptingstandardconsumersetting
- https://docs.microsoft.com/en-us/windows/win32/wmisdk/standard-consumer-classes
- https://docs.microsoft.com/en-us/windows/win32/wmisdk/running-a-script-based-on-an-event

Previous
Access to Microphone Device

Next
Remote DCOM IErtUtil DLL Hijack

By Roberto Rodriguez @Cyb3rWard0g