



FalconForce

**Lifting the veil, a
look at MDE under
the hood**

— TROOPERS
2022



Olaf Hartong

Defensive Specialist @ FalconForce

Detection Engineer and Security Researcher

- Built and/or led Security Operations Centers
- Threat hunting, IR and Compromise assessments

Former documentary photographer

Father of 2 boys

"I like warm hugs"



@olafhartong



github.com/olafhartong



olaf@falconforce.nl



olafhartong.nl / falconforce.nl

What you can expect from this talk

- Microsoft Defender for Endpoint (MDE) capabilities
- What kind of telemetry can you work with
- Where does it get its telemetry from
- Analyzing its configuration
- Log augmentation



Capability outline

What can it do for you?



Microsoft Defender for Endpoint

All-in-one solution for protecting Windows, Mac and Linux Endpoints

- Anti-Virus
- Attack Surface Reduction (ASR)
- Exploit Guard
- Application Control (WDAC)
- EDR Telemetry
- Incident Response
- Software Inventory / Vulnerability Management
- Network Sensor
- DLP

Some parts are also available separately. Defender for Endpoint integrates these parts into a combined product and allows for centralized logging and management.



Anti-Virus Engine

Leverages existing Microsoft Defender Anti-Virus product.

- AV events are logged to M365 Defender Portal.

Signature-based detection (behavior + file characteristics).

Cloud-based detections where samples are uploaded to cloud for analysis and can be executed in a sandbox.

Great research on the signature database by Camille Mougey

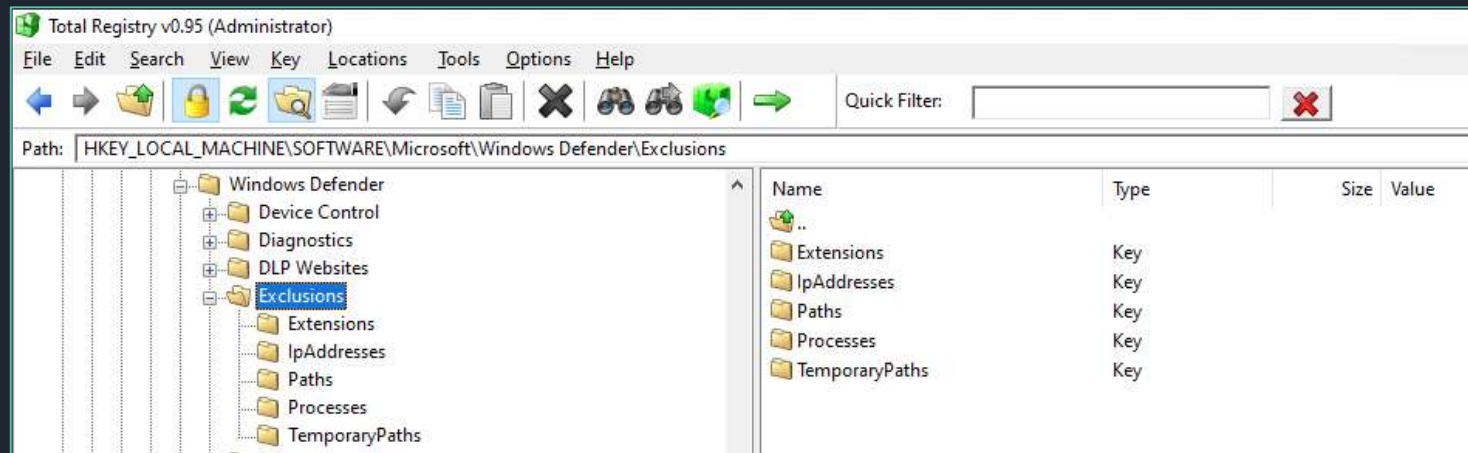
(<https://github.com/commial/experiments/tree/master/windows-defender/VDM>)



Anti-Virus Engine

Exclusions

- Frequently used by attackers to allow their payload to pass, **monitor the registry changes**.
- Process exclusions apply to the children of the listed process. The listed process will **still** be scanned. Unless this file is added to the file exclusion list.
- These exclusions apply **ONLY** for the AV component, features like EDR and ASR still apply.



Anti-Virus Engine

Check what it flags on with DefenderCheck

```
PS C:\Users\olafhartong\Downloads\DefenderCheck> .\DefenderCheck.exe .\DefenderCheck.exe
Target file size: 9216 bytes
Analyzing...

[!] Identified end of bad bytes at offset 0x156C in the original file
File matched signature: "VirTool:MSIL/BytzChk.C!MTB"

00000000  00 74 00 20 00 30 00 78 00 7B 00 30 00 3A 00 58  ·t· ·0·x·{·0·:·X
00000010  00 7D 00 20 00 69 00 6E 00 20 00 74 00 68 00 65  ·}· ·i·n· ·t·h·e
00000020  00 20 00 6F 00 72 00 69 00 67 00 69 00 6E 00 61  · ·o·r·i·g·i·n·a
00000030  00 6C 00 20 00 66 00 69 00 6C 00 65 00 00 65 45  ·l· ·f·i·l·e·eE
00000040  00 78 00 68 00 61 00 75 00 73 00 74 00 65 00 64  ·x·h·a·u·s·t·e·d
00000050  00 20 00 74 00 68 00 65 00 20 00 73 00 65 00 61  · ·t·h·e· ·s·e·a
00000060  00 72 00 63 00 68 00 2E 00 20 00 54 00 68 00 65  ·r·c·h· · ·T·h·e
00000070  00 20 00 62 00 69 00 6E 00 61 00 72 00 79 00 20  · ·b·i·n·a·r·y·
00000080  00 6C 00 6F 00 6F 00 6B 00 73 00 20 00 67 00 6F  ·l·o·o·k·s· ·g·o
00000090  00 6F 00 64 00 20 00 74 00 6F 00 20 00 67 00 6F  ·o·d· ·t·o· ·g·o
000000A0  00 21 00 00 5D 43 00 3A 00 5C 00 50 00 72 00 6F  ·!· ·]C·:·\·P·r·o
000000B0  00 67 00 72 00 61 00 6D 00 20 00 46 00 69 00 6C  ·g·r·a·m· ·F·i·l
000000C0  00 65 00 73 00 5C 00 57 00 69 00 6E 00 64 00 6F  ·e·s·\·W·i·n·d·o
000000D0  00 77 00 73 00 20 00 44 00 65 00 66 00 65 00 6E  ·w·s· ·D·e·f·e·n
000000E0  00 64 00 65 00 72 00 5C 00 4D 00 70 00 43 00 6D  ·d·e·r·\·M·p·C·m
000000F0  00 64 00 52 00 75 00 6E 00 2E 00 65 00 78 00 65  ·d·R·u·n· ·e·x·e
```

Sometimes needs several changes to the source to not get detected anymore.



Attack Surface Reduction (ASR) rules

- ~16 rules to reduce the attack surface of Windows.
- Rules can be enabled and disabled via Reg keys / Group Policy.
- Can be configured to Block or only Audit.
- Events are logged in M365 Advanced Hunting tables.

Safe for most Environments	Environment Specific	Use Caution
• Block untrusted and unsigned processes that run from USB • Block Adobe Reader from creating child processes • Block executable content from email client and webmail • Block JavaScript or VBScript from launching downloaded executable content • Block persistence through WMI event subscription • Block credential stealing from the Windows local security authority subsystem (lsass.exe) • Block Office applications from creating executable content	• Block Office applications from injecting code into other processes • Block Win32 API calls from Office macros • Block all Office applications from creating child processes • Block execution of potentially obfuscated scripts	• Block executable files from running unless they meet a prevalence, age, or trusted list criterion • Use advanced protection against ransomware • Block process creations originating from PSEXEC and WMI commands • Block Office communication applications from creating child processes



Attack Surface Reduction (ASR) rules

First and foremost, enable as much of them as you can, they're quite good and will slow a capable attacker down.

Even in audit only mode they provide great value.

These rules are written in LUA and essentially are signature rules based on regex paths.

The rules are compiled and stored within the Defender AV database.

Camille Mougey decompiled them and made them available here:

<https://github.com/commial/experiments/tree/master/windows-defender/ASR>



<http://youtube.com/watch?v=H3NDuvLqfwE>

Attack Surface Reduction (ASR) rules

The rules (currently) primarily look for file / path names or commandlines, not signer information or other unique attributes. This allows an attacker to bypass them.

```
l_0_1 = (l_0_1 == nil and "" or l_0_1):lower()
if (l_0_1:sub(-20)):match("%.[%w-!+]+$") ~= nil or not "" then
  local l_0_3 = nil
  local l_0_4 = ((mp.PathToWin32Path)((mp.get_contextdata)(mp.CONTEXT_DATA_FILEPATH)) == nil and "" or (mp.PathToWin32Path)((mp.get_contextdata)(mp.CONTEXT_DATA_FILEPATH))):lower()
  local l_0_5 = ((mp.ContextualExpandEnvironmentVariables)("%appdata%") == nil and "" or (mp.ContextualExpandEnvironmentVariables)("%appdata%")):lower()
  local l_0_6 = ((mp.ContextualExpandEnvironmentVariables)("%localappdata%") == nil and "" or (mp.ContextualExpandEnvironmentVariables)("%localappdata%")):lower()
  local l_0_7 = ((mp.ContextualExpandEnvironmentVariables)("%temp%") == nil and "" or (mp.ContextualExpandEnvironmentVariables)("%temp%")):lower()
  local l_0_8 = ((mp.ContextualExpandEnvironmentVariables)("%programdata%") == nil and "" or (mp.ContextualExpandEnvironmentVariables)("%programdata%")):lower()
  local l_0_9 = ((mp.ContextualExpandEnvironmentVariables)("%systemdrive%") == nil and "" or (mp.ContextualExpandEnvironmentVariables)("%systemdrive%")):lower()
  local l_0_10 = ((mp.ContextualExpandEnvironmentVariables)("%systemroot%") == nil and "" or (mp.ContextualExpandEnvironmentVariables)("%systemroot%")):lower()
  if l_0_0[l_0_3] == true then
    if l_0_3 == ".lnk" then
```

```
if l_0_4:find(l_0_5 .. "\\microsoft\\word\\", 1, true) ~= nil then
  return mp.CLEAN
end
if l_0_4:find(l_0_5 .. "\\microsoft\\internet explorer\\quick launch", 1, true) ~= nil then
  return mp.CLEAN
end
if l_0_4:find(l_0_5 .. "\\roaming\\microsoft\\", 1, true) ~= nil then
  return mp.CLEAN
end
```

```
end
if l_0_4 == l_0_5 .. "\\microsoft\\windows\\start menu\\programs\\startup" then
  return mp.CLEAN
end
```



Exploit Guard

Successor to EMET (Enhanced Mitigation Experience Toolkit).

System wide security prevention measures to block certain types of exploits such as buffer overflows.

Many additional features can be enabled per application, for example:

- Block Arbitrary Code.
- Block loading low integrity images (DLLs).
- Disable Direct System Calls.
- Block creation of Child Processes.



Windows Defender Application Control (WDAC)

Used to control which drivers and applications are allowed to run, does not require license!
Successor to AppLocker, available in Windows 10 and up and Server 2016+

Policies can be layered and built to allow on deny based on:

- The codesigning certificate(s)
- Attributes in the PE header
- Reputation in the Microsoft's Intelligent Security Graph
- The path from which the app or file is launched
- The parent process
- The launching identity



EDR Telemetry

Relies on a separate Windows Service, exclusive to MDE called 'Sense' running via MsSense.exe.

Collects relevant data from running system, for example:

- ▶ File Events (File Creation, Deletion).
- ▶ Network Connections.
- ▶ Suspicious API usage such as Reading memory from another process.

All events are logged and stored in 'Advanced Hunting' tables where they can be queried, and custom detection rules can be created to detect unwanted behavior.



EDR Telemetry

Which events are logged is controlled and configured by Microsoft.

- For example: list of registry keys that are monitored is fixed and cannot be extended.
- Focus on events that change the system.

Some events are (heavily) sampled to avoid excessive logging taking place, most notably:

- Network connections.
- File writes.
- Less events are logged from trusted processes (Microsoft-signed).
- Some events such as reading memory from a remote process are limited to LSASS process.

Main data source is Event Tracing for Windows (ETW).

- Over 65 different providers queried.
- This includes 'private' ETW logs, such as Threat Intelligence.



Data Storage

Pay per device / user.

- Includes the storage of generated events.
- Detailed information available for 30 days.
- Timeline/condensed data available for 180 days.

Longer retention possible by copying data to other solutions such as Azure Dataspaces or Azure Sentinel.

- Should be approximately 15-20MB per device per day.



What kind of data can I build detections on hunt with?

OxFF | Ballpit Microsoft 365 Defender

Advanced Hunting

New query + Create new

Schema Functions Queries

Alerts

- AlertInfo
- AlertEvidence

Apps & identities

Email & collaboration

Devices

- DeviceInfo
- DeviceNetworkInfo
- DeviceProcessEvents
- DeviceNetworkEvents
- DeviceFileEvents
- DeviceRegistryEvents
- DeviceLogonEvents
- DeviceImageLoadEvents
- DeviceEvents
- DeviceFileCertificateInfo

Threat & Vulnerability Management

- DeviceTvmSoftwareVulnerabilities
- DeviceTvmSoftwareVulnerabilitiesKB
- DeviceTvmSecureConfigurationAssess...
- DeviceTvmSecureConfigurationAssess...
- DeviceTvmSoftwareInventory
- DeviceTvmSoftwareEvidenceBeta

Advanced Hunting

New query + Create new

Schema Functions Queries

Alerts

- AlertInfo
- AlertEvidence

Apps & identities

Email & collaboration

Devices

- DeviceInfo
- DeviceNetworkInfo
- DeviceProcessEvents
 - Timestamp
 - DeviceId
 - DeviceName
 - ActionType
 - FileName
 - FolderPath
 - SHA1
 - SHA256
 - MD5
 - FileSize
 - ProcessVersionInfoCompanyName
 - ProcessVersionInfoProductName
 - ProcessVersionInfoProductVersion
 - ProcessVersionInfoInternalFileName
 - ProcessVersionInfoOriginalFileName
 - ProcessVersionInfoFileDescription
 - ProcessId

Query

```
1 DeviceProcessEvents
2 | getschema |
```

Getting Started Results

Export 56 items Search 0.016 Low Chart Type Customize columns

ColumnName	ColumnOrdinal	DataType	ColumnType
Timestamp	0	System.DateTime	datetime
DeviceId	1	System.String	string
DeviceName	2	System.String	string
ActionType	3	System.String	string
FileName	4	System.String	string
FolderPath	5	System.String	string
SHA1	6	System.String	string
SHA256	7	System.String	string
MD5	8	System.String	string



Data schema

DeviceEvents
Timestamp
DeviceId
DeviceName
ActionType
FileName
FolderPath
SHA1
SHA256
MD5
FileSize
AccountDomain
AccountName
AccountSid
RemoteUrl
RemoteDeviceName
ProcessId
ProcessCommandLine
ProcessCreationTime
ProcessTokenElevation
LoginId
RegistryKey
RegistryValueName
RegistryValueData
RemoteIP

```
Run query + New Save ✓  
1 DeviceEvents  
2 | summarize count() by ActionType
```

ActionType
AntivirusScanCompleted
ShellLinkCreateFileEvent
AsrOfficeMacroWin32ApiCallsAudited
ProcessPrimaryTokenModified
AntivirusReport
LdapSearch
OpenProcessApiCall
AsrLsassCredentialTheftAudited
DriverLoad
PnpDeviceConnected
ReadProcessMemoryApiCall
NtAllocateVirtualMemoryApiCall
PowerShellCommand
FirewallInboundConnectionBlocked
NtMapViewOfSectionRemoteApiCall
NtAllocateVirtualMemoryRemoteApiCall
CreateRemoteThreadApiCall
ExploitGuardWin32SystemCallBlocked
GetClipboardData
GetAsyncKeyStateApiCall
FirewallOutboundConnectionBlocked
ScreenshotTaken
BrowserLaunchedToOpenUrl
ScheduledTaskCreated
AsrOfficeProcessInjectionAudited
DeviceBootAttestationInfo
AsrExecutableOfficeContentAudited
ScheduledTaskDeleted
ExploitGuardNonMicrosoftSignedAudited
ProcessCreatedUsingWmiQuery
ExploitGuardNonMicrosoftSignedBlocked
ExploitGuardAcgEnforced
ExploitGuardNetworkProtectionAudited
FirewallInboundConnectionToAppBlocked
AsrUntrustedExecutableAudited
UsbDriveMount
WriteProcessMemoryApiCall
AsrOfficeChildProcessAudited
UsbDriveUnmount
ExploitGuardChildProcessAudited
ControlledFolderAccessViolationAudited
UserAccountCreated
AntivirusScanCancelled
ControlledFolderAccessViolationBlocked
MemoryRemoteProtect
AsrExecutableEmailContentAudited
ExploitGuardChildProcessBlocked
AND MUCH, MUCH MORE



Data schema

DeviceEvents
Timestamp
DeviceId
DeviceName
ActionType
FileName
FolderPath
SHA1
SHA256
MD5
FileSize
AccountDomain
AccountName
AccountSid
RemoteUrl
RemoteDeviceName
ProcessId
ProcessCommandLine
ProcessCreationTime
ProcessTokenElevation
LogonId
RegistryKey
RegistryValueName
RegistryValueData
RemoteIP

Run query + New

```
1 DeviceEvents
2 | summarize
```

ActionType
AntivirusScanCompleted
ShellLinkCreateFileEvent
AsrOfficeMacroWin32ApiCallsAudited
ProcessPrimaryTokenModified
AntivirusReport
LdapSearch
CrossProcessApiCall

OxFF | Training Microsoft 365 Defender

Advanced Hunting

New query + Create new

Schema Functions Queries

Alerts

AlertInfo

AlertEvidence

Apps & Identities

IdentityInfo

Devices

DeviceInfo

DeviceNetworkInfo

DeviceProcessEvents

DeviceNetworkEvents

DeviceFileEvents

DeviceRegistryEvents

DeviceLogonEvents

DeviceImageLoadEvents

DeviceEvents

DeviceFileCertificateInfo

Threat & Vulnerability Management

DeviceTimSoftwareVulnerabilities

DeviceTimSoftwareVulnerabilitiesKB

DeviceTimSecureConfigurationAssessment

DeviceTimSecureConfigurationAssessmentKB

DeviceTimSoftwareInventory

DeviceTimSoftwareEvidenceBeta

Run query Save Share link

Query

```
14
15 DeviceProcessEvents
16 | where FileName contains "harmless"
17
```

Getting Started Results

Export Link to incident Take actions 1 of 1 selected Search

Timestamp	DeviceId	DeviceName	ActionType
Apr 11, 2022 9:47:57 AM	8056c32a2e00100...	pc1.falcoferre.local	ProcessC...

DeviceEvents

Description

Multiple event types, including events triggered by security controls such as Windows Defender Antivirus and exploit protection.

Columns

Timestamp

Date and time when the record was generated.

DeviceId

Unique identifier for the device in the service.

DeviceName

Fully qualified domain name (FQDN) of the device.

View all

ActionType values

AccountCheckedForBlankPassword

An account was checked for a blank password.

AntivirusDefinitionsUpdated

Security intelligence updates for Windows Defender Antivirus were applied successfully.

AntivirusDefinitionsUpdateFailed

Security intelligence updates for Windows Defender Antivirus were not applied.

AntivirusDetection

Windows Defender Antivirus detected a threat.

AntivirusEmergencyUpdatesInstalled

Emergency security intelligence updates for Windows Defender Antivirus were applied.

AntivirusError

Windows Defender Antivirus encountered an error while taking action on malware or a potentially unwanted application.

AntivirusMalwareActionFailed

Windows Defender Antivirus attempted to take action on malware or a potentially unwanted application but the action failed.

AntivirusMalwareBlocked

Windows Defender Antivirus blocked files or activity involving malware, potentially unwanted applications or suspicious behavior.

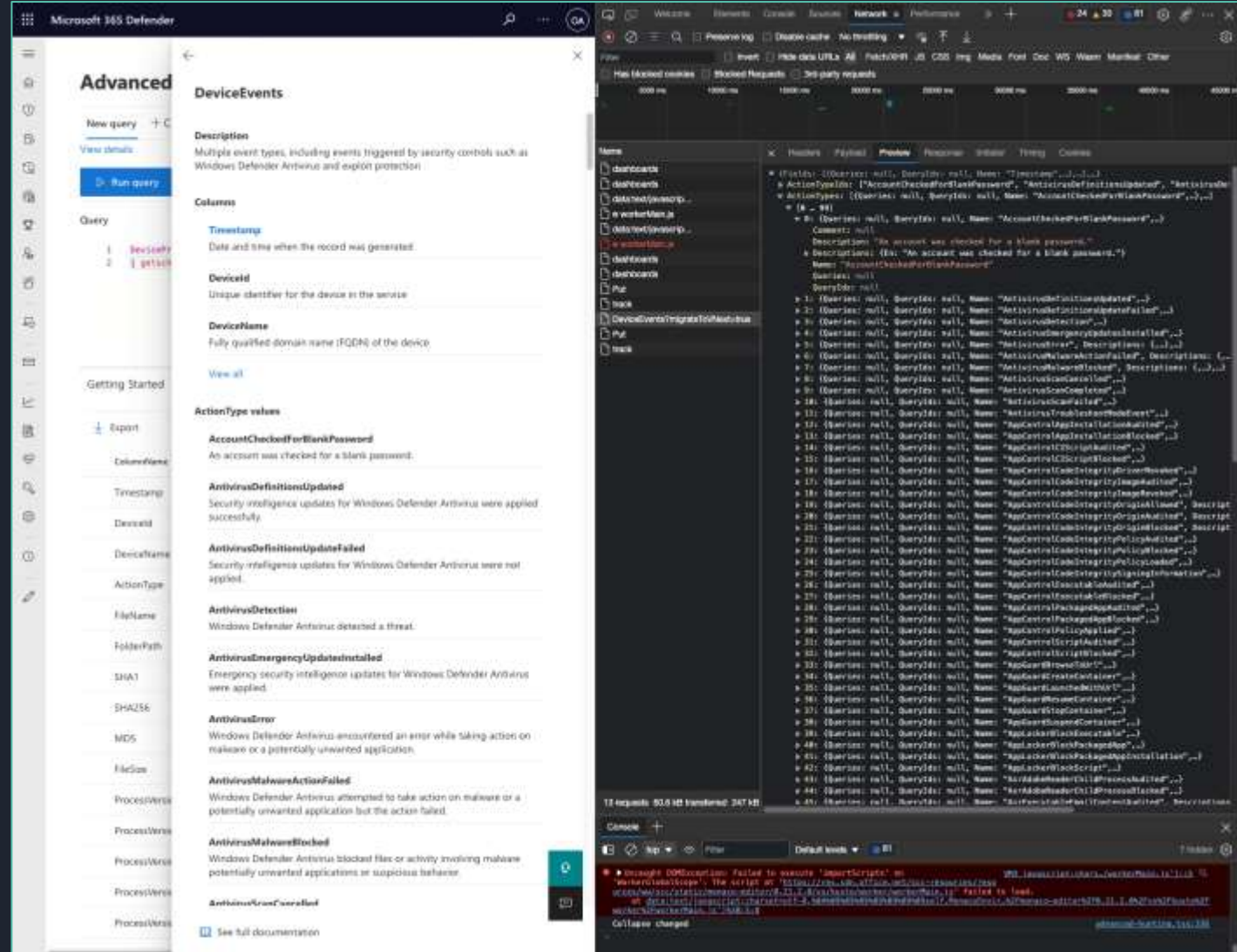
AntivirusScanSucceeded

Snatch them from the portal

```
az login --use-device-code -t [TENANTNAME]
```

```
az account get-access-token --resource  
https://securitycenter.microsoft.com/mtp
```

```
curl -v -H "Authorization: Bearer  
$AZURE_TOKEN" -H 'Content-Type:  
application/json' "https://wdatpprd-  
weu.securitycenter.windows.com/api/ine/hun  
tingservice/documentation/TableDocumentati  
on/DeviceEvents"
```



The screenshot displays the Microsoft 365 Defender Advanced Hunting interface. The left sidebar shows the 'Advanced' section with a 'DeviceEvents' query. The main pane shows the 'DeviceEvents' table with columns: Timestamp, DeviceId, DeviceName, ActionType, FileName, FolderPath, SHA1, SHA256, MD5, FileSize, ProcessName, ProcessPath, ProcessId, ProcessName, ProcessPath, ProcessId. The table lists various events, including 'AccountCheckedForBlankPassword', 'AntivirusDefinitionsUpdated', 'AntivirusDefinitionsUpdateFailed', 'AntivirusDetection', 'AntivirusEmergencyUpdatesInstalled', 'AntivirusError', 'AntivirusMalwareActionFailed', 'AntivirusMalwareBlocked', and 'AntivirusScanFailed'.

The right pane shows the 'DeviceEvents' table with columns: Name, Description, Details, and Columns. The table lists various events, including 'AccountCheckedForBlankPassword', 'AntivirusDefinitionsUpdated', 'AntivirusDefinitionsUpdateFailed', 'AntivirusDetection', 'AntivirusEmergencyUpdatesInstalled', 'AntivirusError', 'AntivirusMalwareActionFailed', 'AntivirusMalwareBlocked', and 'AntivirusScanFailed'.



Snatch them from the portal

```
az login --use-device-code -t [TENANTNAME]
```

```
az account get-access-token --resource  
https://securitycenter.microsoft.com/mtp
```

```
curl -v -H "Authorization: Bearer $AZURE_TOKEN" -X POST  
https://securitycenter.microsoft.com/mtp/application/javascripts/  
on/DeviceEvents
```

ChangeMonitor APP 11:32 AM
Changes to MDE table configuration found:

Column added to MDE schema: AADSignInEventsBeta.RiskEventTypes
New actiontype seen for the first time: DeviceEvents.AppControlCIScriptAudited
New actiontype seen for the first time: DeviceEvents.AppControlCodeIntegrityPolicyAudited
New actiontype seen for the first time: DeviceEvents.AppControlCodeIntegrityPolicyLoaded
New actiontype seen for the first time: DeviceEvents.ExploitGuardChildProcessBlocked
New actiontype seen for the first time: DeviceEvents.SafeDocFileScan
Additional field seen for the first time: DeviceEvents.AppControlCIScriptAudited.Sha1CatalogHash
Additional field seen for the first time: DeviceEvents.AppControlCIScriptAudited.Sha256CatalogHash
Additional field seen for the first time: DeviceEvents.AppControlCodeIntegrityPolicyAudited.PolicyID
Additional field seen for the first time: DeviceEvents.AppControlCodeIntegrityPolicyAudited.PolicyName
Additional field seen for the first time: DeviceEvents.AppControlCodeIntegrityPolicyAudited.Requested Signing Level
Additional field seen for the first time: DeviceEvents.AppControlCodeIntegrityPolicyAudited.Validated Signing Level
Additional field seen for the first time: DeviceEvents.AppControlCodeIntegrityPolicyLoaded.PolicyGuid
Additional field seen for the first time: DeviceEvents.AppControlCodeIntegrityPolicyLoaded.PolicyIdBuffer
Additional field seen for the first time: DeviceEvents.AppControlCodeIntegrityPolicyLoaded.PolicyNameBuffer
Additional field seen for the first time: DeviceEvents.ExploitGuardChildProcessBlocked.ChildCommandLine
Additional field seen for the first time: DeviceEvents.ExploitGuardChildProcessBlocked.IsAudit
Additional field seen for the first time: DeviceEvents.SafeDocFileScan.Container
Additional field seen for the first time: DeviceEvents.SafeDocFileScan.ContainerReason
Additional field seen for the first time: DeviceEvents.SafeDocFileScan.State
Additional field seen for the first time: DeviceEvents.SafeDocFileScan.VerificationResult
Additional field seen for the first time: DeviceNetworkEvents.InboundConnectionAccepted.InitiatingProcessPosixAttachedTerminal

AntivirusMalwareActionFailed
Windows Defender Antivirus attempted to take action on malware or a potentially unwanted application but the action failed.

AntivirusMalwareBlocked
Windows Defender Antivirus blocked file or activity involving malware, potentially unwanted applications or suspicious behavior.

AntivirusScanCancelled
See full documentation

12 requests, 80.0 KB transferred, 347 MB

● [Security Center] failed to execute 'ImportScripts' on 'Windows Defender Antivirus'. The script at 'https://securitycenter.microsoft.com/mtp/application/javascripts/on/DeviceEvents' failed to load. Details: [Security Center] failed to execute 'ImportScripts' on 'Windows Defender Antivirus'. The script at 'https://securitycenter.microsoft.com/mtp/application/javascripts/on/DeviceEvents' failed to load.



Do you need those custom detections ?

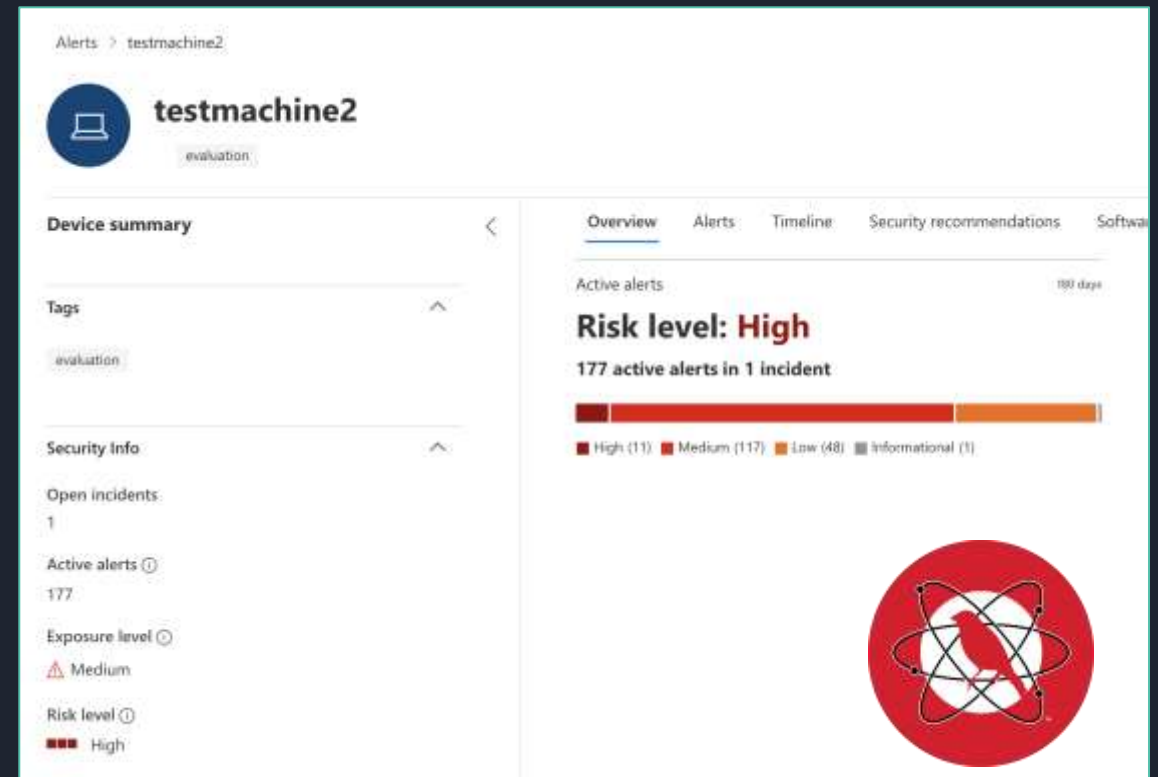
A basic non-scientific test shows you should.

Executed 562 Atomic Red Team (ART) scripts, out of which some failed.

Resulting in 177 alerts based on out of the box rules, so there is a gap.

167 of those alerts are mapped to ATT&CK.

21 of the alerts are mapped to a technique that was not tagged in the ART project

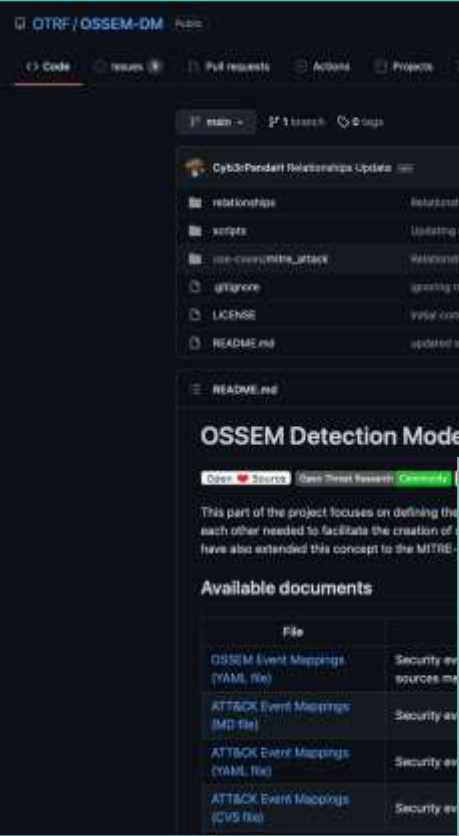


Theoretical telemetry mapping

This can be done based on the schema or the generated data.

Mapping can be done against the OSSEM Detection Model that also is aware of the ATT&CK data sources.

Keep in mind this is biased on two sides: the MITRE mapping as well as the **generated** telemetry.



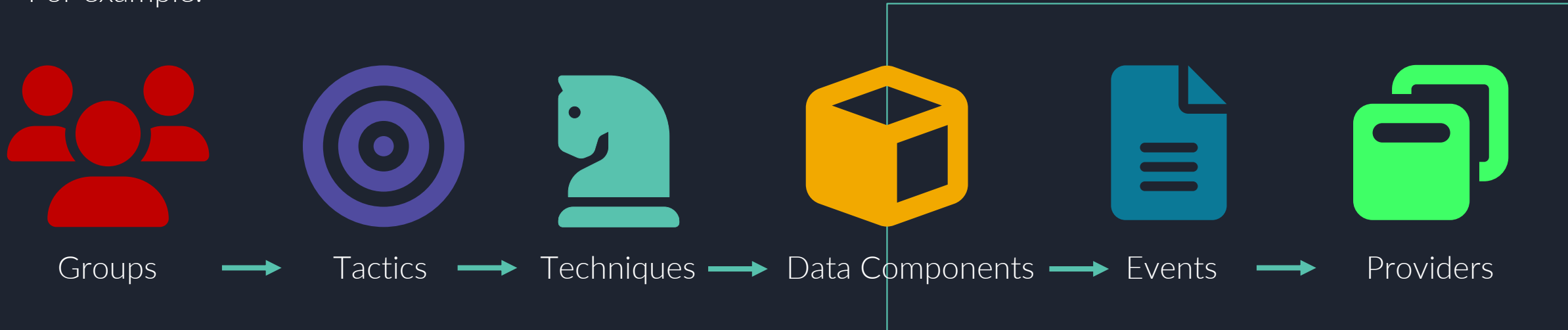
```
name: Process modified Process
contributors:
  - Jose Rodriguez @Cyb3rPandaH
  - Roberto Rodriguez @Cyb3rWard0g
  - Olaf Hartong @olafhartong
attack:
  data_source: Process
  data_component: process modification
behavior:
  source: process
  relationship: modified
  target: process
security_events:
  - event_id: 8
    name: CreateRemoteThread.
    platform: Windows
    audit_category: CreateRemoteThread
    log_channel: Microsoft-Windows-Sysmon/Operational
    log_provider: Microsoft-Windows-Sysmon
  - event_id: CreateRemoteThreadApiCall
    name: CreateRemoteThreadApiCall
    platform: Windows
    audit_category: null
    log_channel: DeviceEvents
    log_provider: Microsoft Defender for Endpoint
references:
notes:
```



Linking data sources > data components > events

Since ATT&CK contains all kinds relations we can start combining sets of relationships with other sets.

For example:



The same can be done for;
tools, detection rules, attack/validation scripts, event fields and much, much more!



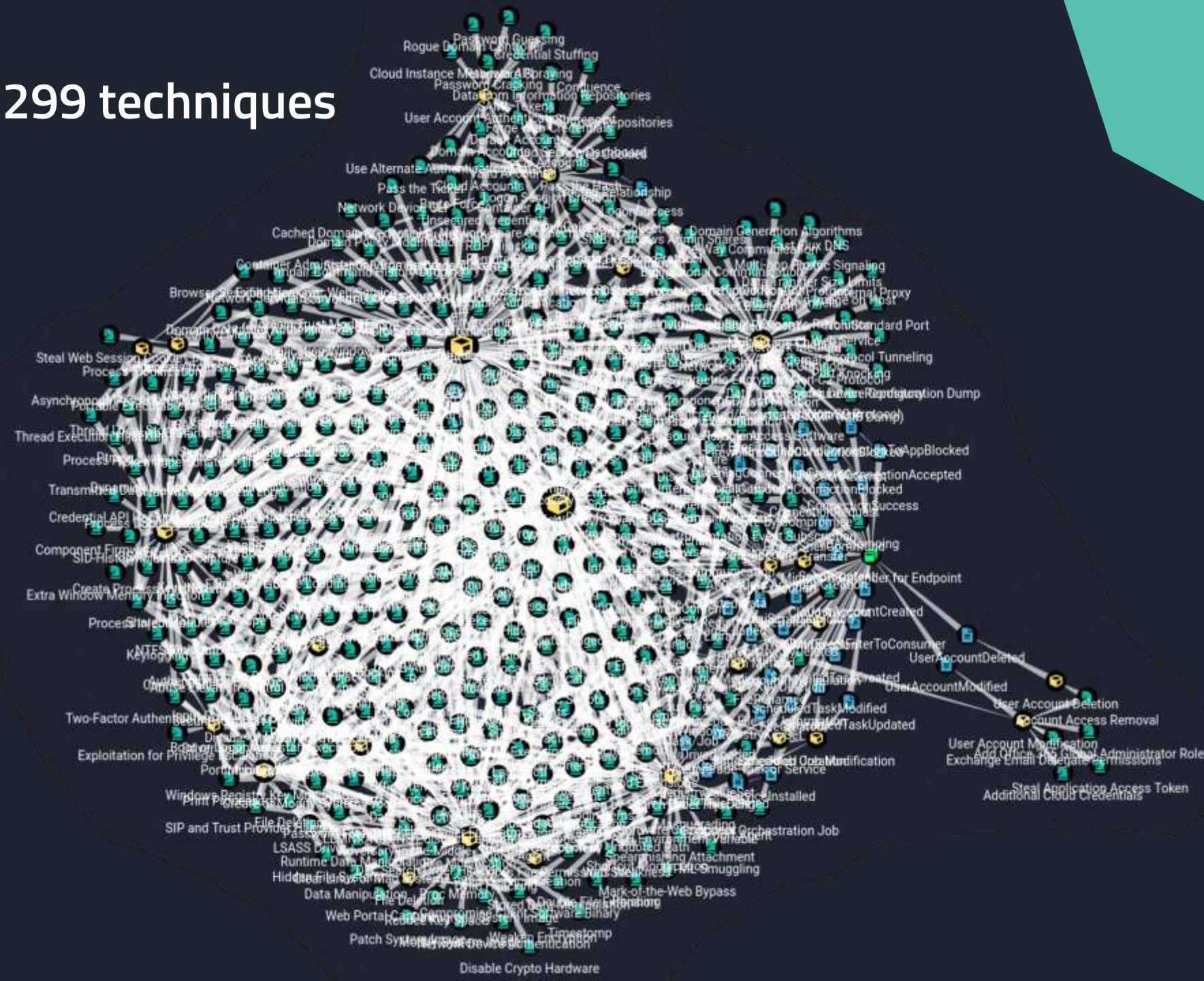
MDE telemetry potential mapping to MITRE ATT&CK

Initial Access 9 techniques	Execution 10 techniques	Persistence 18 techniques	Privilege Escalation 13 techniques	Defense Evasion 32 techniques	Credential Access 15 techniques	Discovery 25 techniques	Lateral Movement 9 techniques	Collection 15 techniques	Command and Control 16 techniques	Exfiltration 8 techniques	Impact 13 techniques
Drive-by Compromise	Command and Scripting Interpreter (5/5)	Account Manipulation (1/3)	Abuse Elevation Control Mechanism (1/1)	Abuse Elevation Control Mechanism (1/1)	Brute Force (4/4)	Account Discovery (3/4)	Exploitation of Remote Services	Archive Collected Data (3/3)	Application Layer Protocol (0/4)	Automated Exfiltration (0/0)	Account Access Removal
Exploit Public-Facing Application	Exploitation for Client Execution	BITS Jobs	Access Token Manipulation (5/5)	Access Token Manipulation (5/5)	Credentials from Password Stores (3/3)	Application Window Discovery	Internal Spearphishing	Audio Capture	Communication Through Removable Media	Data Transfer Size Limits	Data Destruction
External Remote Services	Inter-Process Communication (2/2)	Boot or Logon Autostart Execution (10/10)	Boot or Logon Autostart Execution (10/10)	BITS Jobs	Exploitation for Credential Access	Browser Bookmark Discovery	Lateral Tool Transfer	Automated Collection	Data Encoding (0/2)	Exfiltration Over Alternative Protocol (3/3)	Data Encrypted for Impact
Hardware Additions	Native API	Boot or Logon Initialization Scripts (2/2)	Boot or Logon Initialization Scripts (10/10)	Deobfuscate/Decode Files or Information	Forced Authentication	Cloud Service Dashboard	Remote Service Session Hijacking (1/1)	Clipboard Data	Data Obfuscation (0/3)	Exfiltration Over C2 Channel	Data Manipulation (3/3)
Phishing (0/3)	Scheduled Task/Job (2/2)	Browser Extensions	Boot or Logon Initialization Scripts (2/2)	Direct Volume Access	Forge Web Credentials (2/2)	Cloud Service Discovery	Remote Services (5/5)	Data from Information Repositories (1/1)	Dynamic Resolution (2/3)	Exfiltration Over Other Network Medium (1/1)	Defacement (2/2)
Replication Through Removable Media	Shared Modules	Compromise Client Software Binary	Create or Modify System Process (1/1)	Domain Policy Modification (2/2)	Input Capture (4/4)	Domain Trust Discovery	Replication Through Removable Media	Data from Local System	Encrypted Channel (0/2)	Exfiltration Over Physical Medium (1/1)	Disk Wipe (2/2)
Supply Chain Compromise (0/3)	Software Deployment Tools	Create Account (2/3)	Domain Policy Modification (2/2)	Execution Guardrails (1/1)	Man-in-the-Middle (1/2)	File and Directory Discovery	Software Deployment Tools	Data from Network Shared Drive	Fallback Channels	Exfiltration Over Web Service (2/2)	Endpoint Denial of Service (0/4)
Trusted Relationship	System Services (1/1)	Create or Modify System Process (1/1)	Escape to Host	File and Directory Permissions Modification (1/1)	Modify Authentication Process (2/2)	Network Service Scanning	Taint Shared Content	Data from Removable Media	Ingress Tool Transfer	Scheduled Transfer	Firmware Corruption
Valid Accounts (3/4)	User Execution (2/2)	Event Triggered Execution (11/11)	Event Triggered Execution (11/11)	Hide Artifacts (6/6)	Network Sniffing	Network Share Discovery	Use Alternate Authentication Material (2/4)	Data Staged (2/2)	Multi-Stage Channels		Inhibit System Recovery
	Windows Management Instrumentation	External Remote Services	Exploitation for Privilege Escalation	Hijack Execution Flow (9/9)	OS Credential Dumping (5/6)	Network Sniffing		Email Collection (2/3)	Non-Application Layer Protocol		Network Denial of Service (0/2)
		Hijack Execution Flow (9/9)	Hijack Execution Flow (9/9)	Impair Defenses (5/5)	Steal Application Access Token	Permission Groups Discovery (2/3)		Input Capture (4/4)	Non-Standard Port		Resource Hijacking
		Modify Authentication Process (2/2)	Process Injection (8/8)	Indicator Removal on Host (5/5)	Steal or Forge Kerberos Tickets (0/4)	Process Discovery		Man in the Browser	Protocol Tunneling		Service Stop
		Office Application Startup (6/6)	Scheduled Task/Job (2/2)	Indirect Command Execution	Steal Web Session Cookies	Query Registry		Man-in-the-Middle (1/2)	Proxy (3/4)		System Shutdown/Reboot
		Pre-OS Boot (1/3)	Valid Accounts (3/4)	Process Injection (T1055)	Two-Factor Authentication	Remote System Discovery		Screen Capture	Remote Access Software		
		Scheduled Task/Job (2/2)		Score: 80	DeviceFileEvents_FileModified	Software Discovery (1/1)		Video Capture	Traffic Signaling (1/1)		
		Server Software Component (2/3)		Aggregate Score (average): 92.22	DeviceFileEvents_FileRenamed	System Information Discovery			Web Service (2/3)		
		Traffic Signaling (1/1)		file modification:	DeviceImageLoadEvents_ImageLoaded	System Location Discovery					
		Valid Accounts (3/4)		module load:	DeviceEvents_OpenProcessApiCall	System Network Configuration Discovery (1/1)					
				Obfuscated Files or process access:	DeviceEvents_CreateRemoteThreadApiCall	System Network Connections Discovery					
				OS api execution:	Pre-OS Boot (1/3)	System Owner/User Discovery					
				Process Injection (8/8)	Rogue Domain Controller						
				Rootkit							
				Signed Binary Proxy Execution (11/11)							

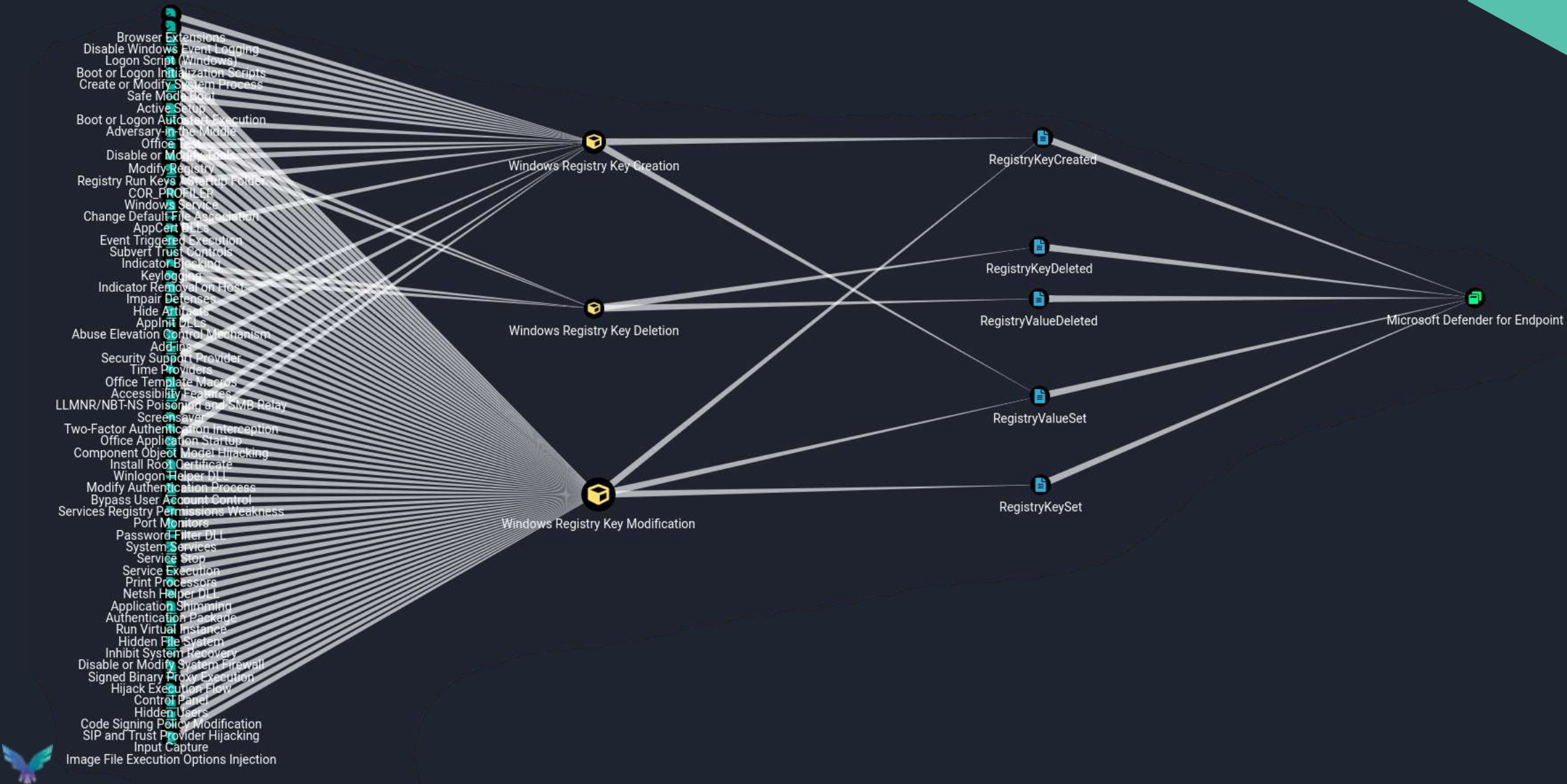
MDE telemetry potential mapping to MITRE ATT&CK

Initial Access 9 techniques	Execution 10 techniques	Persistence 18 techniques	Privilege Escalation 13 techniques	Defense Evasion 32 techniques	Credential Access 15 techniques	Discovery 25 techniques	Lateral Movement 9 techniques	Collection 15 techniques	Command and Control 16 techniques	Exfiltration 8 techniques	Impact 13 techniques
Drive-by Compromise	Command and Scripting Interpreter (5/5)	Account Manipulation (1/3)	Abuse Elevation Control Mechanism (1/1)	Abuse Elevation Control Mechanism (1/1)	Brute Force (4/4)	Account Discovery (3/4)	Exploitation of Remote Services	Archive Collected Data (3/3)	Application Layer Protocol (0/4)	Automated Exfiltration (0/0)	Account Access Removal
Exploit Public-Facing Application	Exploitation for Client Execution	BITS Jobs	Access Token Manipulation (5/5)	Access Token Manipulation (5/5)	Credentials from Password Stores (3/3)	Application Window Discovery	Internal Spearphishing	Audio Capture	Communication Through Removable Media	Data Transfer Size Limits	Data Destruction
External Remote Services	Inter-Process Communication (2/2)	Boot or Logon Autostart Execution (10/10)	Boot or Logon Autostart Execution (10/10)	BITS Jobs	Exploitation for Credential Access	Browser Bookmark Discovery	Lateral Tool Transfer	Automated Collection	Data Encoding (0/2)	Exfiltration Over Alternative Protocol (3/3)	Data Encrypted for Impact
Hardware Additions	Native API	Boot or Logon Initialization Scripts	Boot or Logon Initialization Scripts (10/10)	Deobfuscate/Decode Files or Information	Forced Authentication	Cloud Service Dashboard	Remote Service Session Hijacking	Clipboard Data	Data Obfuscation (0/3)	Exfiltration Over C2 Channel	Data Manipulation (3/3)
Phishing (0/3)						Cloud Service Discovery		Data from Information Sources			Defacement (2/2)
Replication To Removable Medium	Process Injection (8/8)		Host Discovery (5/5)	Indicator Removal On Host (5/5)		Steal or Forge Kerberos Tickets (0/4)		Discovery (2/3)		Exfiltration Over Other Network Medium (1/1)	Disk Wipe (2/2)
Supply Chain Compromise	Scheduled Task/Job (2/2)		Indirect Command Execution	Indirect Command Execution				Process Discovery		Exfiltration Over Physical Medium (1/1)	Endpoint Denial of Service (0/4)
Trusted Relationships	Valid Accounts (3/4)		Process Injection (T1055) (2/5)	Process Injection (T1055) (2/5)		Steal Web Session Cookie		Query Registry		Exfiltration Over Web Service (2/2)	Firmware Corruption
Valid Account Credentials			Score: 80	Score: 80				Remote System Discovery		Scheduled Transfer	Inhibit System Recovery
			Aggregate Score (average): 92.22	Aggregate Score (average): 92.22		Two-Factor Authentication		Software Discovery (1/1)			Network Denial of Service (0/2)
			file modification:	file modification:		DeviceFileEvents_FileModified		System Information Discovery			Resource Hijacking
			Modify Registry	Modify Registry		DeviceFileEvents_FileRenamed		System Location Discovery			Service Stop
			module load:	module load:		DeviceImageLoadEvents_ImageLoaded		System Network Configuration			System Shutdown/Reboot
			process access:	process access:		DeviceEvents_OpenProcessApiCall					
			OS api execution:	OS api execution:		DeviceEvents_CreateRemoteThreadApiCall					
			Pre-OS Boot (1/3)	Pre-OS Boot (1/3)							
			Rogue Domain Controller	Rogue Domain Controller		System network Connections Discovery					
			Rootkit	Rootkit		System Owner/User Discovery					
			Signed Binary Proxy Execution (1/1)	Signed Binary Proxy Execution (1/1)							

Data potential for 299 techniques



Visualizing relationships



Where does it get its telemetry?

This is important to understand bypass and tampering opportunities as well as possible blind spots.

Ekson_T (Aug 24 2020 08:47:19)

ETW Trace Sessions | About

Stop Session

Count: 32 sessions.

Tip: Missing results? Run as SYSTEM to view more sessions

Enabled Provider

Microsoft-Windows-Kernel-Process

Microsoft-Windows-PowerShell

Microsoft-Windows-WMI-Activity

Microsoft-Windows-Sysmon

Microsoft-Windows-DNS-Client

Microsoft-Windows-SMBClient

Microsoft-Windows-SMBServer

Microsoft-Windows-Audit-CVE

Microsoft-Windows-WinINet

Microsoft-Windows-DNS-Client

Microsoft-Windows-DNS-Client

Microsoft-Windows-Kernel-Process

Microsoft-Windows-Kernel-AppCompat

Microsoft-Windows-Application-Experience

Microsoft-Windows-Kernel-PnP

Microsoft-Windows-Diagnostics-Performance

Microsoft-Windows-Kernel-WDI

Microsoft-Windows-UAC-FileVirtualization

Microsoft-Windows-Kernel-WHEA

Microsoft-Windows-Build-RegDll

Microsoft-Windows-DesktopWindowManager-Diag

Kernel Callbacks

The kernel's callback mechanism provides a general way for drivers to request and provide notification when certain conditions are satisfied.

```
mimikatz # !notifprocess
[00] 0xFFFFF8030CB5A2C0 [ntoskrnl.exe + 0x35a2c0]
[01] 0xFFFFF80310AE6DD0 [cng.sys + 0x6dd0]
[02] 0xFFFFF80314805F90 [WdFilter.sys + 0x45f90]
[03] 0xFFFFF8031093B9A0 [ksecdd.sys + 0x1b9a0]
[04] 0xFFFFF80311D58330 [tcpip.sys + 0x48330]
[05] 0xFFFFF80312308A90 [SysmonDrv.sys + 0x8a90]
[06] 0xFFFFF803123ED930 [iorate.sys + 0xd930]
[07] 0xFFFFF80310D2C5C0 [mssecflt.sys + 0x2c5c0]
[08] 0xFFFFF80310A6A050 [CI.dll + 0x7a050]
[09] 0xFFFFF80312E0AFB0 [dxgkrnl.sys + 0xafb0]
[10] 0xFFFFF8031346A420 [vm3dmp.sys + 0xa420]
[11] 0xFFFFF80314543CE0 [peauth.sys + 0x43ce0]
```

```
mimikatz # !notifreg
[00] 0xFFFFF80312309EA0 [SysmonDrv.sys + 0x9ea0]
[01] 0xFFFFF803147F7820 [WdFilter.sys + 0x37820]
[02] 0xFFFFF80310D2F190 [mssecflt.sys + 0x2f190]
[03] 0xFFFFF8030CDCAF50 [ntoskrnl.exe + 0x5caf50]
```

```
mimikatz # !notifimage
[00] 0xFFFFF803148068E0 [WdFilter.sys + 0x468e0]
[01] 0xFFFFF8031230E3C0 [SysmonDrv.sys + 0xe3c0]
[02] 0xFFFFF80310D2C8A0 [mssecflt.sys + 0x2c8a0]
[03] 0xFFFFF80313DAEB20 [ahcache.sys + 0x1eb20]
```

```
mimikatz # !notifthread
[00] 0xFFFFF80314807680 [WdFilter.sys + 0x47680]
[01] 0xFFFFF803148073E0 [WdFilter.sys + 0x473e0]
[02] 0xFFFFF80312308240 [SysmonDrv.sys + 0x8240]
[03] 0xFFFFF80310D24000 [mssecflt.sys + 0x24000]
[04] 0xFFFFF803144B1060 [mmcss.sys + 0x1060]
```

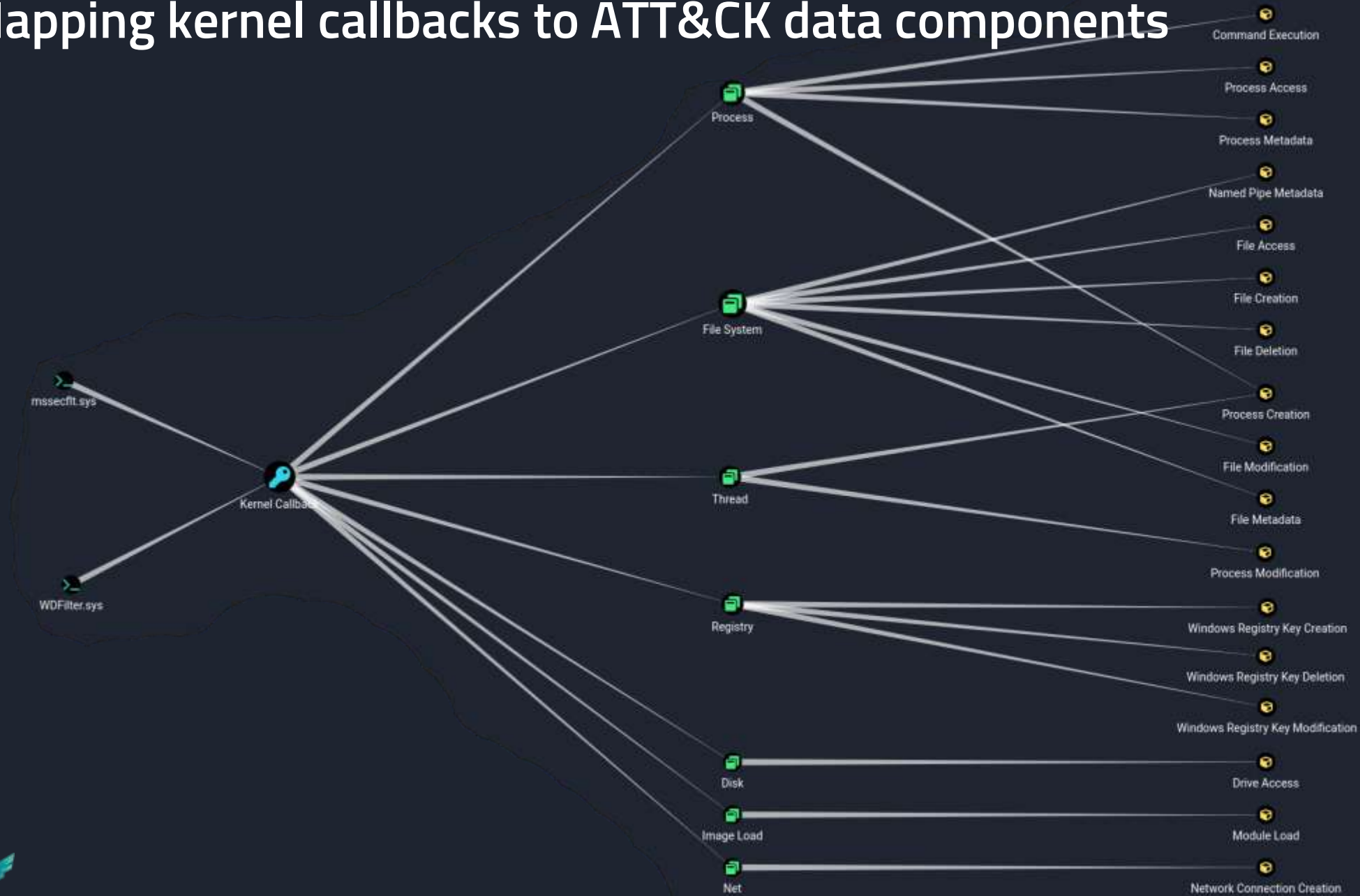


Kernel Callbacks

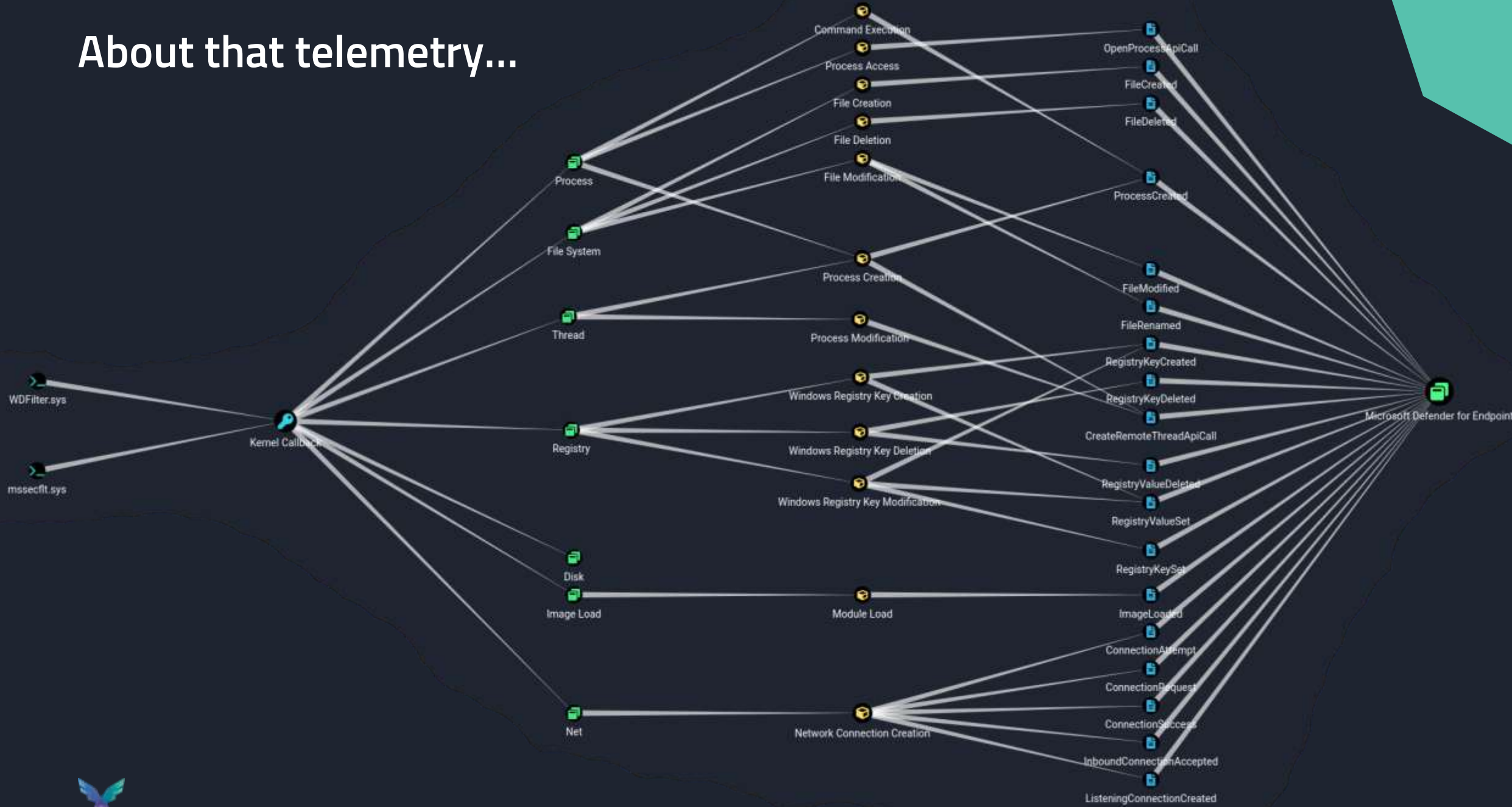
```
* Process
  * Callback [type 3] - Handle 0xFFFFB20FABC50910 (@ 0xFFFFB20FABC50930)
    PreOperation : 0xFFFFF80310D19A60 [mssecflt.sys + 0x19a60]
  * Callback [type 3] - Handle 0xFFFFB20FAE0300E0 (@ 0xFFFFB20FAE030100)
    PreOperation : 0xFFFFF80314803D90 [WdFilter.sys + 0x43d90]
  * Callback [type 1] - Handle 0xFFFFB20FABE42290 (@ 0xFFFFB20FABE422B0)
    PreOperation : 0xFFFFF80312305080 [SysmonDrv.sys + 0x5080]
    PostOperation : 0xFFFFF803123092C0 [SysmonDrv.sys + 0x92c0]
  Open      - 0xFFFFF8030CEB5830 [ntoskrnl.exe + 0x6b5830]
  Close     - 0xFFFFF8030CEE48B0 [ntoskrnl.exe + 0x6e48b0]
  Delete    - 0xFFFFF8030CE1A210 [ntoskrnl.exe + 0x61a210]
  Security  - 0xFFFFF8030CE691A0 [ntoskrnl.exe + 0x6691a0]
```



Mapping kernel callbacks to ATT&CK data components



About that telemetry...



Event Tracing for Windows

Event Tracing for Windows (ETW) provides a mechanism to trace and log events that are raised by **user-mode applications** and **kernel-mode drivers**.

ETW is implemented in the Windows operating system and provides a fast, reliable, and versatile set of event tracing features. Its architecture consists of three layers;

- Event providers
- Event consumers
- Event tracing sessions

Great reference material by Matt Graeber:

<https://blog.palantir.com/tampering-with-windows-event-tracing-background-offense-and-defense-4be7ac62ac63>

<https://posts.specterops.io/data-source-analysis-and-dynamic-windows-re-using-wpp-and-tracelogging-e465f8b653f7>



MsSense.exe ETW Providers

MsSense is one of the core components of MDE that routes the telemetry which it gathers in its own set of providers.

Curious about the traces it utilizes I had a look at the trace logging metadata with a script created by Matt Graeber.

```
PS C:\Users\olafhartong\Downloads> $Result = Get-TraceLoggingMetadata -Path 'C:\Program Files\Windows Defender Advanced Threat Protection\MsSense.exe'
PS C:\Users\olafhartong\Downloads> $Result.Providers
```

ProviderGUID	ProviderName	ProviderGroupGUID
-----	-----	-----
65a1b6fc-4c24-59c9-e3f3-ad11ac510b41	Microsoft.Windows.Sense.Client	5ecb0bac-b930-47f5-a8a4-e8253529edb7
c60418cc-7e07-400f-ae3b-d521c5dbd96f	Microsoft.Windows.Sense.GeneratedETW	d0b1a44b-5ab3-4ff2-bb52-c2bb980ef8f3
1dc742c2-0e76-5490-e1b5-8ddb4982ff77	Microsoft.Windows.Sense.SensorHub	c5a3a379-e5b9-43da-9175-509abfce2cc7
cb2ff72d-d4e4-585d-33f9-f3a395c40be7	Microsoft.Windows.Sense.CyberEvents	541dae91-cc3c-5807-b064-c2561c16d7e8
b3861234-4273-58c5-545b-8b3611343471	Microsoft.Windows.Sense.CyberEvents	
001600f9-311e-5cff-2d59-ee6d065ad02b	Microsoft.Windows.Sense.Ndr	4f50731a-89cf-4782-b3e0-dce8c90476ba
450bba94-53ce-54e6-d150-9636aceafb86	Microsoft.Windows.Sense.SenseIR	
f68c769c-cc20-502e-aee3-115c2eda66f7	Microsoft.Windows.Sense.CollectionEtw	d0b1a44b-5ab3-4ff2-bb52-c2bb980ef8f3



MsSense.exe ETW data

The traced events are stored into a SQLite database in a protected folder on the file system. The table name used is AsimovEvents.

Asimov was the code name in 2014 for the Unified **Telemetry** Client, which is now deprecated and is replaced by the DiagTrack agent.

On regular intervals the contents of the database gets uploaded and the data gets flushed..

```
Directory of C:\ProgramData\Microsoft\Windows Defender Advanced Threat Protection\Cyber

15/07/2021  11:53    <DIR>          .
15/07/2021  11:53    <DIR>          ..
04/05/2022  20:30             35.651.584 EventStore.db
20/04/2022  18:43             32.768 EventStore.db-shm
04/05/2022  20:33             1.048.576 EventStore.db-wal
               3 File(s)             36.732.928 bytes
```



Tracing these providers

Curious to see what these providers contained I fired up Sealightner to trace these file to a file.

Sealightner is highly configurable and can subscribe to multiple providers at once, [user](#) and [kernel](#) traces.

Outputs to Stdout, JSON file, or Windows Event Log

<https://github.com/pathtofile/Sealightner>

Primarily built for research, if you want to use custom ETW events for monitoring SilkETW is probably more suited.



```
c:\tools\service>sealightner.exe config.json
Session Name: MDE-traces
Outputs: file
User Provider: {65a1b6fc-4c24-59c9-e3f3-ad11ac510b41}
  Trace Name: Microsoft.Windows.Sense.Client
  Keywords: All
  No event filters
User Provider: {c60418cc-7e07-400f-ae3b-d521c5dbd96f}
  Trace Name: Microsoft.Windows.Sense.GeneratedETW
  Keywords: All
  No event filters
User Provider: {1dc742c2-0e76-5490-e1b5-8ddb4982ff77}
  Trace Name: Microsoft.Windows.Sense.SensorHub
  Keywords: All
  No event filters
User Provider: {cb2ff72d-d4e4-585d-33f9-f3a395c40be7}
  Trace Name: Microsoft.Windows.Sense.CyberEvents
  Keywords: All
  No event filters
User Provider: {b3861234-4273-58c5-545b-8b3611343471}
  Trace Name: Microsoft.Windows.Sense.CyberEvents
  Keywords: All
  No event filters
User Provider: {314159be-26a1-cf39-e3f3-ad11ac510b41}
  Trace Name: Microsoft.Windows.SenseNdr
  Keywords: All
  No event filters
User Provider: {001600f9-311e-5cff-2d59-ee6d065ad02b}
  Trace Name: Microsoft.Windows.Sense.Ndr
  Keywords: All
  No event filters
User Provider: {450bba94-53ce-54e6-d150-9636aceafb86}
  Trace Name: Microsoft.Windows.Sense.SenseIR
  Keywords: All
  No event filters
User Provider: {f0ff433a-b5a0-4899-a81d-0b5088a96d04}
  Trace Name: Microsoft.Windows.Sense.SenseCm
  Keywords: All
  No event filters
User Provider: {f68c769c-cc20-502e-aee3-115c2eda66f7}
  Trace Name: Microsoft.Windows.Sense.CollectionEtw
  Keywords: All
  No event filters
User Provider: {7af898d7-7e0e-518d-5f96-b1e79239484c}
  Trace Name: Microsoft.Windows.Defender
  Keywords: All
  No event filters
User Provider: {e2cdb57-b2a5-570a-969b-ef80adc0b915}
  Trace Name: Microsoft.Windows.Sec.Driver
  Keywords: All
  No event filters
Starting User Trace...
-----
```

Protected providers

Some of these providers are protected.

You at least to run as a Protected Process Light (PPL) to be able to access them.

With the Microsoft-Windows-Threat-Intelligence provider you also need a Microsoft Early Launch AntiMalware driver (ELAM) that also runs as PPL.

This can be achieved, albeit a bit cumbersome

Great tool and blog by Patrick Hogan;

<https://github.com/pathtofile/PPLRunner>

<https://blog.tofile.dev/2020/12/16/elam.html>

This is also how my colleague Gijs found a spoofing vulnerability

<https://medium.com/falconforce/debugging-the-undebuggable-and-finding-a-cve-in-microsoft-defender-for-endpoint-ce36f50bb31>



Sample trace

```
▼ {header: {...}, properties: {...}, property_types: {...}}
▼ header: {activity_id: "{00000000-0000-0000-0000-000000000000}", event_flags: 577, event_id: 0, event_name: "CyberSecurity", event_opcode: 0, event_version: 0, process_id: 3028, provider_name: "Micro...", ...}
  activity_id: "{00000000-0000-0000-0000-000000000000}"
  event_flags: 577
  event_id: 0
  event_name: "CyberSecurity"
  event_opcode: 0
  event_version: 0
  process_id: 3028
  provider_name: "Microsoft.Windows.Sense.CyberEvents"
  task_name: "CyberSecurity"
  thread_id: 4596
  timestamp: "2022-04-20 09:15:11Z"
  trace_name: "Microsoft.Windows.Sense.CyberEvents"
▼ properties: {EventMetadata: "{\\\"EventType\\\":\\\"GenericEtwEvent\\\",\\\"Truncation\\\":0,\\\"RuleId\\\":\\\"{674630AF-0442-4BC1-9C42-ECBB62CAD5CC}\\\"}\", IsEventCompressed: 0, PartA_iKey: \"P-WDATP\", SenseEpoch: 1391..., ...}
  EventMetadata: "{\\\"EventType\\\":\\\"GenericEtwEvent\\\",\\\"Truncation\\\":0,\\\"RuleId\\\":\\\"{674630AF-0442-4BC1-9C42-ECBB62CAD5CC}\\\"}"
  IsEventCompressed: 0
  PartA_iKey: "P-WDATP"
  SenseEpoch: 13914262
  SenseSeqNum: 15845
  events: "rQkLAQ9HZW5lcmJlRXR3RXZlbnQKAakPR2VuZXJpY0V0d0V2ZW50ygrFBgnGCs5VvZfykpXsAcoRBcDb17sCJM+ZAKSzgQFmrL+ktYOH1qLKAADGD6ewjpPykpXsAcIUAMIZAMYj3cegl/KSlewBASrJBg5wb3dlcnNoZWxsLmV4ZckLDnBvd2Vyc2h1bGw...
  id: "5562633633664632663363362634365662633661622633632643633333336136666433363436653353366139373662313236653462316434633666373233633616534336437366500313623636343623133363431231363435666...
▼ property_types: {EventMetadata: "STRINGA", IsEventCompressed: "UINT8", PartA_iKey: "STRINGW", SenseEpoch: "UINT32", SenseSeqNum: "UINT32", events: "STRINGA", id: "ERROR"}
  EventMetadata: "STRINGA"
  IsEventCompressed: "UINT8"
  PartA_iKey: "STRINGW"
  SenseEpoch: "UINT32"
  SenseSeqNum: "UINT32"
  events: "STRINGA"
  id: "ERROR"
```

Base64?





2

What is the binary jibberish?

The data is serialized with Bond.

Bond is a cross-platform framework for working with [schematized data](#). It supports cross-language de/serialization and powerful generic mechanisms for efficiently manipulating data. Bond is broadly used at Microsoft in most of their services.

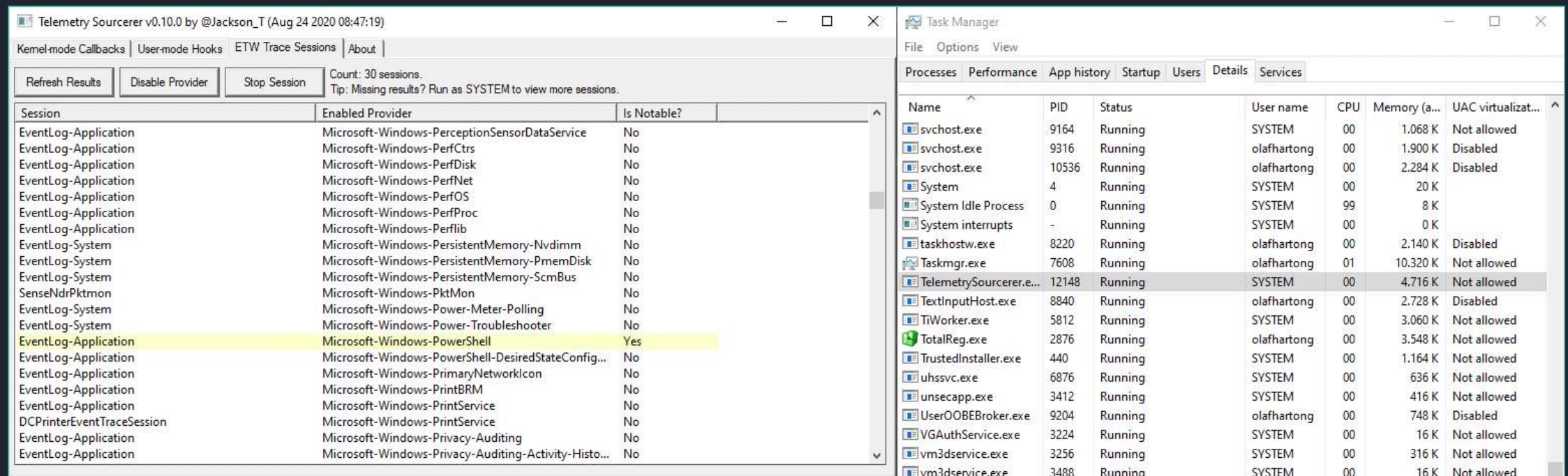
So far I have not found the schema's for these streams.

Next question is where is that data coming from, it clearly looks like PowerShell event logging.



Where is the data coming from?

No direct subscription for anything other than the EventLog service



The image shows two windows side-by-side. The left window is 'Telemetry Sourcerer v0.10.0 by @Jackson_T (Aug 24 2020 08:47:19)'. It has tabs for 'Kernel-mode Callbacks', 'User-mode Hooks', 'ETW Trace Sessions', and 'About'. The 'ETW Trace Sessions' tab is active, showing a table of sessions. The right window is 'Task Manager' with the 'Processes' tab active, showing a list of running processes.

Session	Enabled Provider	Is Notable?
EventLog-Application	Microsoft-Windows-PerceptionSensorDataService	No
EventLog-Application	Microsoft-Windows-PerfCtrs	No
EventLog-Application	Microsoft-Windows-PerfDisk	No
EventLog-Application	Microsoft-Windows-PerfNet	No
EventLog-Application	Microsoft-Windows-PerfOS	No
EventLog-Application	Microsoft-Windows-PerfProc	No
EventLog-Application	Microsoft-Windows-Perflib	No
EventLog-System	Microsoft-Windows-PersistentMemory-Nvdim	No
EventLog-System	Microsoft-Windows-PersistentMemory-PmemDisk	No
EventLog-System	Microsoft-Windows-PersistentMemory-ScmBus	No
SenseNdrPktmon	Microsoft-Windows-PktMon	No
EventLog-System	Microsoft-Windows-Power-Meter-Polling	No
EventLog-System	Microsoft-Windows-Power-Troubleshooter	No
EventLog-Application	Microsoft-Windows-PowerShell	Yes
EventLog-Application	Microsoft-Windows-PowerShell-DesiredStateConfig...	No
EventLog-Application	Microsoft-Windows-PrimaryNetworkIcon	No
EventLog-Application	Microsoft-Windows-PrintBRM	No
EventLog-Application	Microsoft-Windows-PrintService	No
DCPrinterEventTraceSession	Microsoft-Windows-PrintService	No
EventLog-Application	Microsoft-Windows-Privacy-Auditing	No
EventLog-Application	Microsoft-Windows-Privacy-Auditing-Activity-Histo...	No

Name	PID	Status	User name	CPU	Memory (a...	UAC virtualizat...
svchost.exe	9164	Running	SYSTEM	00	1.068 K	Not allowed
svchost.exe	9316	Running	olafhartong	00	1.900 K	Disabled
svchost.exe	10536	Running	olafhartong	00	2.284 K	Disabled
System	4	Running	SYSTEM	00	20 K	
System Idle Process	0	Running	SYSTEM	99	8 K	
System interrupts	-	Running	SYSTEM	00	0 K	
taskhostw.exe	8220	Running	olafhartong	00	2.140 K	Disabled
Taskmgr.exe	7608	Running	olafhartong	01	10.320 K	Not allowed
TelemetrySourcerer.e...	12148	Running	SYSTEM	00	4.716 K	Not allowed
TextInputHost.exe	8840	Running	olafhartong	00	2.728 K	Disabled
TiWorker.exe	5812	Running	SYSTEM	00	3.060 K	Not allowed
TotalReg.exe	2876	Running	olafhartong	00	3.548 K	Not allowed
TrustedInstaller.exe	440	Running	SYSTEM	00	1.164 K	Not allowed
uhssvc.exe	6876	Running	SYSTEM	00	636 K	Not allowed
unsecapp.exe	3412	Running	SYSTEM	00	416 K	Not allowed
UserOOBEBroker.exe	9204	Running	olafhartong	00	748 K	Disabled
VGAAuthService.exe	3224	Running	SYSTEM	00	16 K	Not allowed
vm3dservice.exe	3256	Running	SYSTEM	00	316 K	Not allowed
vm3dservice.exe	3488	Running	SYSTEM	00	16 K	Not allowed

So is MDE **also** making use of the regular EventLogs??



DiagTrack

MDE piggybacks of the Diagtrack service to get most of the ETW event telemetry. This service uses the DiagTrack-Listener subscription. MDE is not subscribing to all these providers itself.

By default, only Local Administrators, Performance Log Users, and services running as LocalSystem, LocalService, NetworkService can control trace sessions and consume event data.

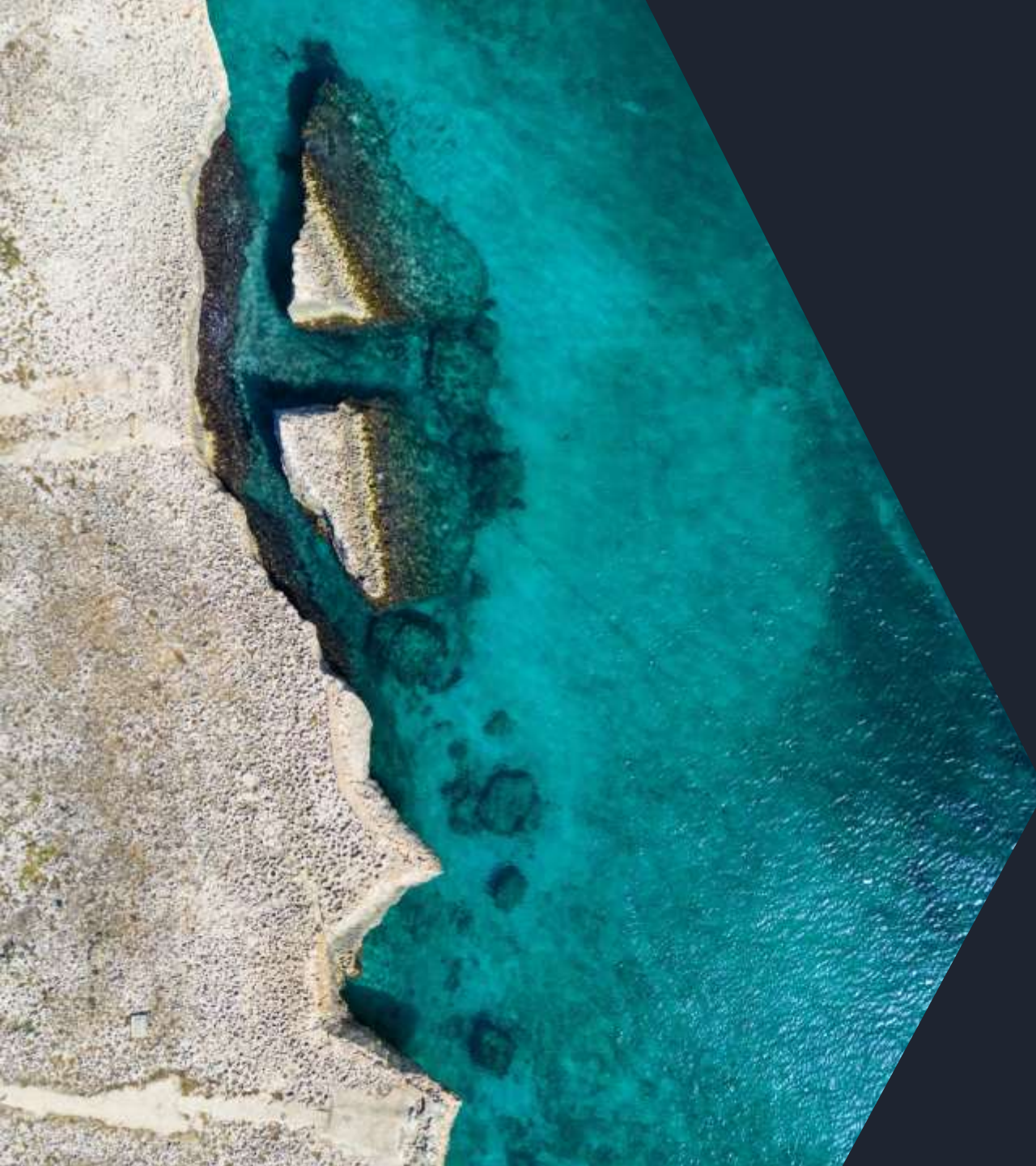
Since MDE uses the MsSense service, which runs as System this is fine.

Looking into this service I learnt this service is not protected. When you stop the DiagTrack service, there is no telemetry sent to the cloud anymore.

```
C:\Users\falconforce>sc qprotection diagtrack  
[SC] QueryServiceConfig2 SUCCESS  
SERVICE diagtrack PROTECTION LEVEL: NONE.
```



```
C:\Users\olafhartong>sc queryex diagtrack  
  
SERVICE_NAME: diagtrack  
        TYPE               : 10  WIN32_OWN_PROCESS  
        STATE                : 4   RUNNING  
                                (STOPPABLE, NOT_PAUSABLE, ACCEPTS_PRESHUTDOWN)  
        WIN32_EXIT_CODE       : 0   (0x0)  
        SERVICE_EXIT_CODE    : 0   (0x0)  
        CHECKPOINT           : 0x0  
        WAIT_HINT            : 0x0  
        PID                  : 2812  
        FLAGS                 :
```



Configuration

MDE Configuration

Like any product MDE also requires a configuration to know what to log.

This configuration is maintained by Microsoft and is downloaded from the internet on a regular basis.

It is stored on the box, in a non-clear text format.
Additionally it is signed and not easily tampered with.

The exact details are up to you to find out ;) (*sorry, not sorry*)



Configuration item examples

- Telemetry sources (ETW providers, Registry Keys etc.)
- Exclusions and Filters (for example; extensions, process names, certificate signatures)
- Capping (global and per event distinct field combination)
- Dynamic data collection
- Agent configuration
- Quotas (volumetric per time period)



Configuration stats

- ~ 70k lines of JSON
- ~ 65 ETW Providers utilized
- ~ 500 registry paths monitored
- ~ 60 data collection commands that fire frequently
- Different settings for high latency environments
- Elevated child process recording quotas for scripting tools and browsers



Configuration - ETW Providers (a selection)

Generic ETW CreateFile Pattern

Microsoft-Windows-ThreatIntelligence

< Very interesting provider, only for AV/EDRs

Microsoft-Windows-DNS-Client

Microsoft.Web.Platform

Microsoft-Windows-Win32k

Microsoft-Antimalware-Scan-Interface

Microsoft-Antimalware-UacScan

Microsoft-Windows-TCPIP

Microsoft-Windows-WMI-Activity

Powershell cmdlets

< We've just seen these events

Microsoft-Windows-AppLocker

Microsoft-Windows-CodeIntegrity

Microsoft.Windows.OLE.Clipboard

Microsoft-Windows-RemoteDesktopServices-RdpCoreTS

Microsoft-Windows-RPC

Microsoft-Windows-SEC

SecureETW

< What would this be?



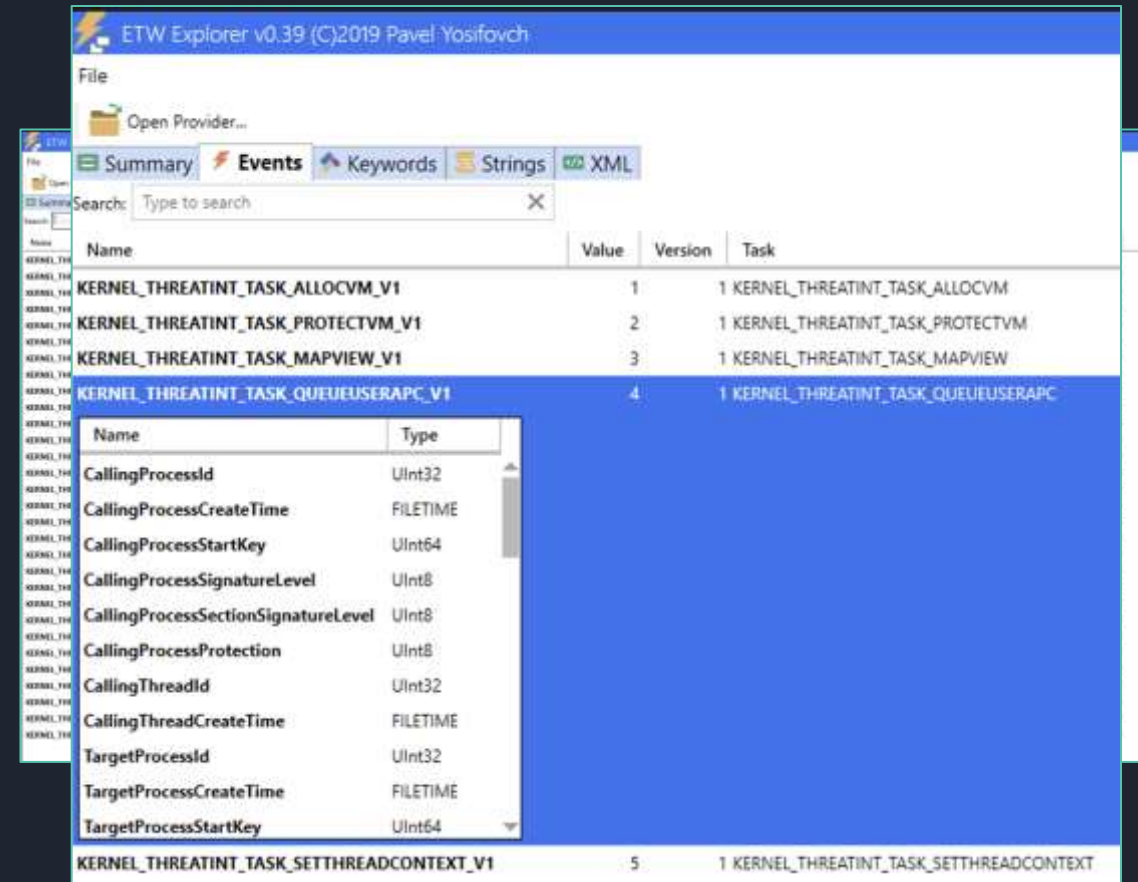
Microsoft-Windows-ThreatIntelligence

Windows native provider, only available to MS
Authorized AV and EDR vendors.

Provides very rich telemetry into all kinds of API calls
like;

- kernel32!VirtualAllocEx or ntdll!NtAllocateVirtualMemory
- kernel32!QueueUserAPC or ntdll!NtQueueApcThread
- kernel32!ReadProcessMemory or
ntdll!NtReadVirtualMemory
- kernel32!SuspendThread or ntdll!NtSuspendThread
- kernel32!SetThreadContext or ntdll!NtSetContextThread
- ntdll!NtLoadDriver

And more



Name	Value	Version	Task
KERNEL_THREATINT_TASK_ALLOCVM_V1	1	1	KERNEL_THREATINT_TASK_ALLOCVM
KERNEL_THREATINT_TASK_PROTECTVM_V1	2	1	KERNEL_THREATINT_TASK_PROTECTVM
KERNEL_THREATINT_TASK_MAPVIEW_V1	3	1	KERNEL_THREATINT_TASK_MAPVIEW
KERNEL_THREATINT_TASK_QUEUEUSERAPC_V1	4	1	KERNEL_THREATINT_TASK_QUEUEUSERAPC
KERNEL_THREATINT_TASK_SETTHREADCONTEXT_V1	5	1	KERNEL_THREATINT_TASK_SETTHREADCONTEXT

Name	Type
CallingProcessId	UInt32
CallingProcessCreateTime	FILETIME
CallingProcessStartKey	UInt64
CallingProcessSignatureLevel	UInt8
CallingProcessSectionSignatureLevel	UInt8
CallingProcessProtection	UInt8
CallingThreadId	UInt32
CallingThreadCreateTime	FILETIME
TargetProcessId	UInt32
TargetProcessCreateTime	FILETIME
TargetProcessStartKey	UInt64



SecureETW

Listed in the configuration with the following
ProviderGuid: {54849625-5478-4994-A5BA-3E3B0328C30D}

```
PS C:\Users\olafhartong> logman query providers "{54849625-5478-4994-A5BA-3E3B0328C30D}"
```

Provider	GUID
Microsoft-Windows-Security-Auditing	{54849625-5478-4994-A5BA-3E3B0328C30D}

Value	Keyword	Description
0x8000000000000000	Security	Security

Value	Level	Description
0x04	win:Informational	Information

The command completed successfully.

Also known as [Microsoft-Windows-Security-Auditing](#)



What does that config look like?

```
{
  "eventId": 4624,
  "aggregation": {
    "type": "NoAggregation"
  },
  "id": "{25FC59D8-3DE9-41EA-A4D6-AE68D5131ECC}",
  "name": "Logon event",
  "version": "1",
  "filters": {
    "expressionType": "Operator",
    "operator": "Not",
    "expressions": [
      {
        // < some SID filters
      }
    ]
  }
},
```

```
"capping": {
  "globalCapping": {
    "capping": 1000
  },
  "localCapping": [
    {
      "id": "LogonLocalCapping",
      "expirationPeriodInHours": 1,
      "fields": [
        {
          "fieldName": "TargetUserName"
        },
        {
          "fieldName": "TargetDomainName"
        },
        {
          "fieldName": "TargetUserSid"
        },
        {
          "fieldName": "LogonType"
        },
        {
          "fieldName": "IpAddress"
        },
        {
          "fieldName": "TargetLogonId"
        }
      ],
      "capping": 1
    }
  ]
},
```

```
"properties": [
  {
    "source": "SubjectUserSid",
    "type": "SID"
  },
  {
    "source": "SubjectUserName",
    "type": "UNICODESTRING",
    "scrubType": "User",
    "scrubMethod": "Simple",
    "scrubProfile": 514
  },
  {
    "source": "SubjectDomainName",
    "type": "UNICODESTRING",
    "scrubType": "Domain",
    "scrubMethod": "Simple",
    "scrubProfile": 516
  },
  {
    "source": "SubjectLogonId",
    "type": "HEXINT64"
  },
  {
    "source": "TargetUserSid",
    "type": "SID",
    "transformer": "ExtractUser",
    "targetFieldName": "TargetAccountEntity",
    "transformerValues": [
      "SID"
    ]
  },
  {
    "source": "TargetUserName",
    "type": "UNICODESTRING",
    "scrubType": "User",
    "scrubMethod": "Simple",
    "scrubProfile": 514
  },
  {
    "source": "TargetDomainName",
    "type": "UNICODESTRING",
    "scrubType": "Domain",
    "scrubMethod": "Simple",
    "scrubProfile": 516
  },
  < and much more
],
```



So, which other EventIDs is it looking for

Currently, the following Events are traced from the Security log:

```
olafhartong mde_config 19:22 $config.configTypes.SensorHubConfig.GenericEtwConfiguration.GenericEtwConfig | Where-Object Name -Match "SecureETW" | select -ExpandProperty Rules
```

eventId	name	id
5058	Persistent cryptographic key operation.	{0051E74D-9FD8-46D3-9DEB-87D89A6AD527}
5059	Persistent cryptographic key export.	{008D33DB-2237-4325-BE48-24F236509208}
4670	Taking Ownership on File from TrustedInstaller	{56EC7AA1-767F-41AD-89C0-B729EFEBE111}
4670	Taking Ownership on MDE Key	{34588649-FDD3-411A-8DEC-6DBBD9131609}
4664	Hardlink Create Audit Event	{63EC7AA1-767F-41AD-89C0-B729EFEBE199}
4907	Sense tampering through object sac1 change	{8A3FC3B0-489B-4E30-AE8C-6239E1AEAE4C}
4697	A service was installed	{18AE52D8-3DE2-41EA-A8e1-AE68D6254ADE}
4624	Logon event	{25FC59D8-3DE9-41EA-A4D6-AE68D5131ECC}
4625	An account failed to log on	{69EA1768-2BAE-45C7-92B7-3F1CE3227148}
4698	A scheduled task was created	{98AE59D8-3DE9-41EA-A8e1-AE68D5254ADE}
4699	A scheduled task was deleted	{03D77EE2-A9FC-4095-811A-586D7D7D1183}
4702	A scheduled task was updated	{2256CB9A-3117-436B-AC84-AD9D36C945B3}
4720	A user account was created	{820D9CBE-975D-42F7-925D-F1314A714572}
6416	Plug and Play event	{8FC5FF9B-B703-4E18-9973-4EE7E9381B00}
5024	Firewall service started	{D5805090-E42C-47B9-9C67-5AF43976331B}
5025	Firewall service stopped	{42ccc346-ee75-4ad6-a834-102e4f74a42b}
5031	Firewall app blocked from listening	{f6c36f47-f999-4162-b373-3011e29a3d7a}
5157	Firewall has blocked a connection outbound	{E818DB90-F7E5-4361-BD88-22D782316AD4}
5157	Firewall has blocked a connection inbound	{7BA4CED0-A91D-491B-B1D0-C3E0ABA1D6BB}
5376	Credman - Credentials Backup	{C7AA73A5-5526-4391-9E18-D42442E4F085}
5379	Credman - Read Credentials	{32D127B2-BEB3-407A-B44C-626AABE16926}
5380	Vault Credential - Find Credential	{348BE356-46FC-4201-B7D1-B0B61007EE84}
5381	Vault Credential - Enumerate Credentials	{028E9574-5F5F-4A85-9598-ACF5E594C351}
5382	Vault Credential - Get Unique Credential	{EF70EE34-531A-4CAF-A27D-420A33CE9DE5}
4648	Logon using explicit credentials	{C0A6D471-F8B4-4F85-B30F-E05147AE5BA5}
4719	System Audit Policy was changed	{7D29E5C5-8E9C-4386-9DEB-0782E635D0C2}
4724	An Attempt was made to reset an account password	{C51A1874-FF0F-4EA5-BC1E-217BA4F10778}
4726	A user account was deleted	{9C88B3E6-D1D3-4A4C-93AC-F8102CC170C1}
4732	A member was added to a security-enabled local group	{70D2074F-B7B2-4D47-852C-B5E0A332C92D}
4731	Local group created	{cbfc31ce-24be-483f-be0d-99fee5133951}
4726	A user account was deleted	{BE56A97E-E1D7-4E23-9DE7-8D2E0D6F2467}
4733	Local group removed	{2f0f972d-7117-41aa-a432-1469b4eb30c0}
4734	Local group deleted	{41fd378a-d621-4eac-acfd-9d4a2e4a0a3f}
4738	A user account was changed	{F7CE3108-BDCB-4C0B-9E77-F1F2AAFEA80E}
4732	A member was added to a security-enabled local group	{E0FE2E6D-D983-4D80-8D06-80E7D2B7AC89}
6423	Forbidden installation (PNP Audit)	{10FE2E6D-D983-4D80-8D06-80E7D2B7AC89}
4798	User's local group membership was enumerated	{6ef3cfd1-a874-4a15-9dc5-43f8c19537bc}
4799	Security-enabled local group membership was enumerated	{9b9ca1b2-ab46-46f6-848f-30f37f057c28}



Mapping EventIDs to name and Audit Category

eventId	MDE-Name	AuditCategory	AuditSubCategory
5058	Persistent cryptographic key operation.	System	Other System Events
5059	Persistent cryptographic key export.	System	Other System Events
4670	Taking Ownership on File from TrustedInstaller	Policy Change	Authorization Policy Change
4670	Taking Ownership on MDE Key	Policy Change	Authorization Policy Change
4664	Hardlink Create Audit Event	Object Access	File System
4907	Sense tampering through object sacl change	Policy Change	Audit Policy Change
4697	A service was installed	System	Security System Extension
4624	Logon event	Logon/Logoff	Logon
4625	An account failed to log on	Logon/Logoff	Logon
4698	A scheduled task was created	Object Access	Other Object Access Events
4699	A scheduled task was deleted	Object Access	Other Object Access Events
4702	A scheduled task was updated	Object Access	Other Object Access Events
4720	A user account was created	Account Management	User Account Management
6416	Plug and Play event	Detailed Tracking	Plug and Play Events
5024	Firewall service started	System	Other System Events
5025	Firewall service stopped	System	Other System Events
5031	Firewall app blocked from listening	Object Access	Filtering Platform Connection
5157	Firewall has blocked a connection outbound	Object Access	Filtering Platform Connection
5157	Firewall has blocked a connection inbound	Object Access	Filtering Platform Connection
5376	Credman - Credentials Backup	Account Management	User Account Management
5379	Credman - Read Credentials	Logon/Logoff	Other Logon/Logoff Events
5380	Vault Credential - Find Credential	Logon/Logoff	Other Logon/Logoff Events
5381	Vault Credential - Enumerate Credentials	Logon/Logoff	Other Logon/Logoff Events
5382	Vault Credential - Get Unique Credential	Logon/Logoff	Other Logon/Logoff Events
4648	Logon using explicit credentials	Logon/Logoff	Logon
4719	System Audit Policy was changed	Policy Change	Audit Policy Change
4724	An Attempt was made to reset an account password	Account Management	User Account Management
4726	A user account was deleted	Account Management	User Account Management
4732	A member was added to a security-enabled local group	Account Management	Security Group Management
4731	Local group created	Account Management	Security Group Management
4726	A user account was deleted	Account Management	User Account Management
4733	Local group removed	Account Management	Security Group Management
4734	Local group deleted	Account Management	Security Group Management
4738	A user account was changed	Account Management	User Account Management
4732	A member was added to a security-enabled local group	Account Management	Security Group Management
6423	Forbidden installation (PNP Audit)	Detailed Tracking	Plug and Play Events
4798	User's local group membership was enumerated	Account Management	User Account Management
4799	Security-enabled local group membership was enumerated	Account Management	Security Group Management



Microsoft Audit Policy settings

Audit policy settings determine whether the operating system generates audit events when certain tasks are performed.

These settings can be configured on 4 levels:

- No Auditing (0)
- Success (1)
- Failure (2)
- Success and Failure (3)



Are all these events available on all machines?

eventId	MDE-Name	AuditCategory	AuditSubCategory	Required setting	Win10 default	Default Ok?
5058	Persistent cryptographic key operation.	System	Other System Events	3	3	TRUE
5059	Persistent cryptographic key export.	System	Other System Events	3	3	TRUE
4670	Taking Ownership on File from TrustedInstaller	Policy Change	Authorization Policy Change	1	0	FALSE
4670	Taking Ownership on MDE Key	Policy Change	Authorization Policy Change	1	0	FALSE
4664	Hardlink Create Audit Event	Object Access	File System	1	0	FALSE
4907	Sense tampering through object sacd change	Policy Change	Audit Policy Change	1	1	TRUE
4697	A service was installed	System	Security System Extension	1	0	FALSE
4624	Logon event	Logon/Logoff	Logon	1	3	TRUE
4625	An account failed to log on	Logon/Logoff	Logon	2	3	TRUE
4698	A scheduled task was created	Object Access	Other Object Access Events	1	0	FALSE
4699	A scheduled task was deleted	Object Access	Other Object Access Events	1	0	FALSE
4702	A scheduled task was updated	Object Access	Other Object Access Events	1	0	FALSE
4720	A user account was created	Account Management	User Account Management	1	1	TRUE
6416	Plug and Play event	Detailed Tracking	Plug and Play Events	1	0	FALSE
5024	Firewall service started	System	Other System Events	1	3	TRUE
5025	Firewall service stopped	System	Other System Events	1	3	TRUE
5031	Firewall app blocked from listening	Object Access	Filtering Platform Connection	2	0	FALSE
5157	Firewall has blocked a connection outbound	Object Access	Filtering Platform Connection	2	0	FALSE
5157	Firewall has blocked a connection inbound	Object Access	Filtering Platform Connection	2	0	FALSE
5376	Credman - Credentials Backup	Account Management	User Account Management	1	1	TRUE
5379	Credman - Read Credentials	Logon/Logoff	Other Logon/Logoff Events	2	0	FALSE
5380	Vault Credential - Find Credential	Logon/Logoff	Other Logon/Logoff Events	2	0	FALSE
5381	Vault Credential - Enumerate Credentials	Logon/Logoff	Other Logon/Logoff Events	2	0	FALSE
5382	Vault Credential - Get Unique Credential	Logon/Logoff	Other Logon/Logoff Events	2	0	FALSE
4648	Logon using explicit credentials	Logon/Logoff	Logon	1	3	TRUE
4719	System Audit Policy was changed	Policy Change	Audit Policy Change	1	1	TRUE
4724	An Attempt was made to reset an account password	Account Management	User Account Management	3	1	FALSE
4726	A user account was deleted	Account Management	User Account Management	1	1	TRUE
4732	A member was added to a security-enabled local group	Account Management	Security Group Management	1	1	TRUE
4731	Local group created	Account Management	Security Group Management	1	1	TRUE
4726	A user account was deleted	Account Management	User Account Management	1	1	TRUE
4733	Local group removed	Account Management	Security Group Management	1	1	TRUE
4734	Local group deleted	Account Management	Security Group Management	1	1	TRUE
4738	A user account was changed	Account Management	User Account Management	1	1	TRUE
4732	A member was added to a security-enabled local group	Account Management	Security Group Management	1	1	TRUE
6423	Forbidden installation (PNP Audit)	Detailed Tracking	Plug and Play Events	1	0	FALSE
4798	User's local group membership was enumerated	Account Management	User Account Management	1	1	TRUE
4799	Security-enabled local group membership was enumerated	Account Management	Security Group Management	1	1	TRUE



So, we seem to be having some blind spots

Fortunately, the MDE team tries to help you a bit here.

They'll enable some of the settings when you install the agent.

eventId	MDE-Name	AuditCategory	AuditSubCategory	Required Setting	Win10 Default	Win10 + Defender	DefaultOk?	PostDefenderInstall
4670	Taking Ownership on File from TrustedInstaller	Policy Change	Authorization Policy Change	1	0	0	FALSE	FALSE
4670	Taking Ownership on MDE Key	Policy Change	Authorization Policy Change	1	0	0	FALSE	FALSE
4664	Hardlink Create Audit Event	Object Access	File System	1	0	3	FALSE	TRUE
4697	A service was installed	System	Security System Extension	1	0	3	FALSE	TRUE
4698	A scheduled task was created	Object Access	Other Object Access Events	1	0	3	FALSE	TRUE
4699	A scheduled task was deleted	Object Access	Other Object Access Events	1	0	3	FALSE	TRUE
4702	A scheduled task was updated	Object Access	Other Object Access Events	1	0	3	FALSE	TRUE
6416	Plug and Play event	Detailed Tracking	Plug and Play Events	1	0	3	FALSE	TRUE
5031	Firewall app blocked from listening	Object Access	Filtering Platform Connection	2	0	0	FALSE	FALSE
5157	Firewall has blocked a connection outbound	Object Access	Filtering Platform Connection	2	0	0	FALSE	FALSE
5157	Firewall has blocked a connection inbound	Object Access	Filtering Platform Connection	2	0	0	FALSE	FALSE
5379	Credman - Read Credentials	Logon/Logoff	Other Logon/Logoff Events	2	0	0	FALSE	FALSE
5380	Vault Credential - Find Credential	Logon/Logoff	Other Logon/Logoff Events	2	0	0	FALSE	FALSE
5381	Vault Credential - Enumerate Credentials	Logon/Logoff	Other Logon/Logoff Events	2	0	0	FALSE	FALSE
5382	Vault Credential - Get Unique Credential	Logon/Logoff	Other Logon/Logoff Events	2	0	0	FALSE	FALSE
4724	An Attempt was made to reset an account password	Account Management	User Account Management	3	1	3	FALSE	TRUE
6423	Forbidden installation (PNP Audit)	Detailed Tracking	Plug and Play Events	1	0	3	FALSE	TRUE



So, we seem to be having some possible blind spots

- However, the categories that are producing a larger volume of telemetry are untouched to not interfere with the log ingestion volume on your SIEM.
- These settings are **not** documented in the MDE documentation and might be overwritten by Group Policy settings.
- Make sure to [check your GPOs](#) and enable the events you care about. Otherwise there will be no telemetry **AND** no alerts on these events.



PowerShell script to check your environment

I've created an ugly script to check all your GPOs are set properly.

Obviously some are layered so make sure to check that too.

The script relies on the Remote Server Administration Tools (RSAT).

It's available on my GitHub:

<https://github.com/olafhartong/MDE-AuditCheck>



```
PS C:\Users\olafhartong.HATCHERY\Desktop> .\MDE-AuditCheck.ps1
This script checks the Group Policies for Audit settings
Next it makes sure all categories that can impact MDE functionality are set properly
There is a total of 10 GPOs.

The following GPOs contain Audit settings:
Audit Settings: Workstations Enhanced Auditing Policy
Audit Settings: Default Domain Controllers Policy
Audit Settings: Servers Enhanced Auditing Policy
Audit Settings: Terrible Idea

Out of those, the following GPOs have potential blind spots due to lacking audit settings
GPO: Workstations Enhanced Auditing Policy
  Authorization Policy Change - Not Set
GPO: Default Domain Controllers Policy
  Audit Logon - Not Set
  Authorization Policy Change - Not Set
  Audit Security Group Management - Not Set
  Audit User Account Management - Not Set
  Audit PNP Activity - Not Set
  Audit Other Logon/Logoff Events - Not Set
  Audit File System - Not Set
  Audit Filtering Platform Connection - Not Set
  Audit Other Object Access Events - Not Set
  Audit Audit Policy Change - Not Set
  Audit Other System Events - Not Set
  Audit Security System Extension - Not Set
GPO: Servers Enhanced Auditing Policy
  Authorization Policy Change - Not Set
GPO: Terrible Idea
  Audit Logon - Expected setting is 3, current setting is: 0
  Authorization Policy Change - Not Set
  Audit Security Group Management - Expected setting is 1 or 3, current setting is: 0
  Audit User Account Management - Expected setting is 1 or 3, current setting is: 0
  Audit PNP Activity - Expected setting is 1 or 3, current setting is: 0
  Audit Other Logon/Logoff Events - Expected setting is 2 or 3, current setting is: 0
  Audit File System - Expected setting is 1 or 3, current setting is: 0
  Audit Filtering Platform Connection - Expected setting is 2 or 3, current setting is: 0
  Audit Other Object Access Events - Expected setting is 1 or 3, current setting is: 0
  Audit Audit Policy Change - Not Set
  Audit Other System Events - Expected setting is 1 or 3, current setting is: 0
  Audit Security System Extension - Expected setting is 1 or 3, current setting is: 0
```

Choose your next chapter(s)



Telemetry
augmentation
within MDE

+

Sysmon vs
MDE

OR

Blind spots
due to timing
problems in
MDE



MDE telemetry augmentation

Using WDAC



The ImageLoadEvents are heavily/randomly sampled

This makes the table very unreliable. No events does not mean it did not happen.

```
18 DeviceImageLoadEvents
19 | where Timestamp > ago(1d)
20 | where DeviceName startswith "pc1."
21 | summarize count() by FileName, InitiatingProcessFileName, InitiatingProcessCommandLine
22 | where FileName in~ ("clr.dll", "clrjit.dll", "mscorlib.dll", "mscorlib.ni.dll")
```

Getting Started Results

No results found in the specified time frame.



WDAC logging in MDE

As we've previously established, the DeviceEvents table holds a wealth of information.

This includes several CodeIntegrity events which are gathered from the EventLog.

What if we create an Audit policy on binaries we care about?

AppControlCodeIntegrityPolicyAudited

Application control detected a code integrity policy violation.

AppControlCodeIntegrityDriverRevoked

Application control found a driver with a revoked certificate.

AppControlCodeIntegrityImageAudited

Application control detected an executable file that violated code integrity policies.

AppControlCodeIntegrityImageRevoked

Application control found an executable file with a revoked certificate.

AppControlCodeIntegrityOriginAllowed

Application control allowed a file due to its good reputation (ISG) or installation source (managed installer).

AppControlCodeIntegrityOriginAudited

Application control would have blocked a file due to its bad reputation (ISG) or installation source (managed installer) if the policy was enforced.

AppControlCodeIntegrityOriginBlocked

Application control blocked a file due to its bad reputation (ISG) or installation

policy violation.

AppControlCodeIntegrityPolicyBlocked

Application control blocked a code integrity policy violation.

AppControlCodeIntegrityPolicyLoaded

An application control code integrity policy was loaded.

AppControlCodeIntegritySigningInformation

Application control signing information was generated.



What if we (ab)use WDAC to help us out ?

```

11 DeviceEvents
12 | where Timestamp > ago(7d)
13 | where ActionType == "AppControlCodeIntegrityPolicyAudited"
14 | where DeviceName startswith "pcl." and isnotempty( FileName)
15 | extend ValidatedSigningLevel=extractjson("$.Validated Signing Level",AdditionalFields, typeof(string))
16 | summarize count() by FileName,FolderPath ,InitiatingProcessFileName,InitiatingProcessVersionInfoOriginalFileName,InitiatingProcessCommandLine,ValidatedSigningLevel, InitiatingProcessParentFileName
17 | where FileName in~ {"clr.dll","clrjit.dll","mscorlib.dll","mscorlib.ni.dll","mscorlib.ni.dll"}
18 ..

```

Getting Started

Results

Export

70 items

Search

0:0.203

Low

Chart Type

Customize columns

☐	FileName	FolderPath	InitiatingProcessFileName	InitiatingProcessVersion...	InitiatingProcessCommandL...	ValidatedSigningLevel	InitiatingProcessParentFile...	count
☐	mscorlib.dll	\Device\HarddiskVolum...	gocasey.exe	AutomatedExecution.exe	"GoCasey.exe"	Windows-signed	explorer.exe	1
☐	mscorlib.dll	\Device\HarddiskVolum...	gocasey.exe	AutomatedExecution.exe	"GoCasey.exe"	Microsoft-signed	explorer.exe	1
☐	mscorlib.dll	\Device\HarddiskVolum...	gocasey.exe	AutomatedExecution.exe	"GoCasey.exe"	Microsoft-signed	explorer.exe	1
☐	mscorlib.dll	\Device\HarddiskVolum...	gocasey.exe	AutomatedExecution.exe	"GoCasey.exe"	Microsoft-signed	explorer.exe	1
☐	mscorlib.dll	\Device\HarddiskVolum...	gravedigger.exe	AutomatedExecution.exe	"Gravedigger.exe"	Windows-signed	explorer.exe	1
☐	mscorlib.dll	\Device\HarddiskVolum...	gravedigger.exe	AutomatedExecution.exe	"Gravedigger.exe"	Microsoft-signed	explorer.exe	1
☐	mscorlib.dll	\Device\HarddiskVolum...	gravedigger.exe	AutomatedExecution.exe	"Gravedigger.exe"	Microsoft-signed	explorer.exe	1
☐	mscorlib.dll	\Device\HarddiskVolum...	gravedigger.exe	AutomatedExecution.exe	"Gravedigger.exe"	Microsoft-signed	explorer.exe	1



Creating a policy

Policies can be enabled in Audit and/or Enforcement mode, this can be used in mixed mode too.

You can create a base policy and add supplemental policies, which makes it easier to maintain and do testing.

This section only highlights a small set of the capabilities, WDAC is **very** powerfull



Creating a policy - Files

The original filename is a great asset for renamed / copied files and fortunately can be used in a WDAC policy as well.

Keep in mind the OriginalFileName in the PE header *CAN be different* than the filename of the binary.

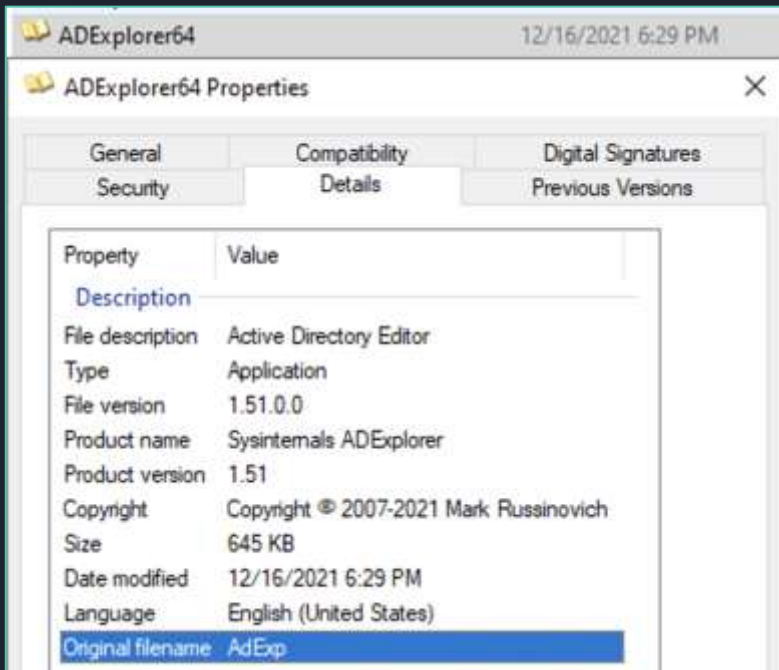


Table 2. Windows Defender Application Control policy - file rule levels

Rule level	Description
Hash	Specifies individual hash values for each discovered binary. This is the most specific level, and requires more effort to maintain the current product versions' hash values. Each time a binary is updated, the hash value changes, therefore requiring a policy update.
FileName	Specifies the original filename for each binary. Although the hash values for an application are modified when updated, the file names are typically not. This level offers less specific security than the hash level, but it doesn't typically require a policy update when any binary is modified.
FilePath	Beginning with Windows 10 version 1903, this level allows binaries to run from specific file path locations. More information about FilePath level rules can be found below.



Creating a policy

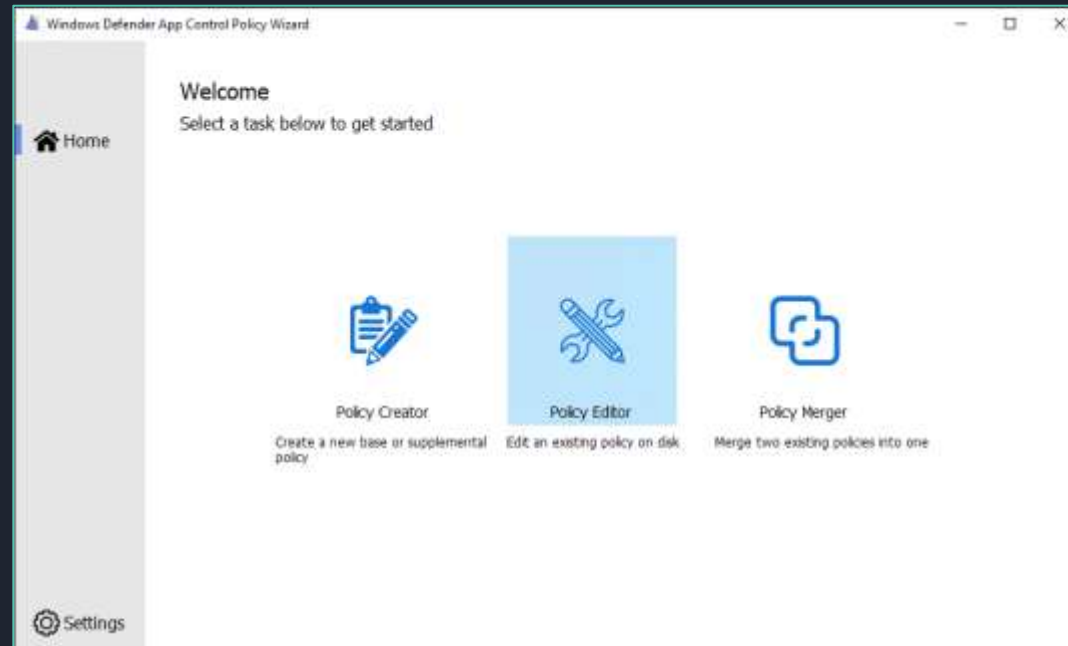
```
9      <Rule>
10        <Option>Enabled:Audit Mode</Option>
11      </Rule>
12      <Rule>
13        <Option>Enabled:Unsigned System Integrity Policy</Option>
14      </Rule>
15      <Rule>
16        <Option>Enabled:Advanced Boot Options Menu</Option>
17      </Rule>
18      <Rule>
19        <Option>Enabled:Update Policy No Reboot</Option>
20      </Rule>
21      <!-- Enable .NET hardening auditing-->
22      <Rule>
23        <Option>Enabled:Dynamic Code Security</Option>
24      </Rule>
25    </Rules>
26    <!-- EKUs -->
27    <EKUs />
28    <!-- File Rules -->
29    <FileRules>
30      <Allow ID="ID_ALLOW_A_1_0" FriendlyName="" FileName="*" />
31      <Allow ID="ID_ALLOW_A_2_0" FriendlyName="" FileName="*" />
32      <Deny ID="ID_DENY_D_0_0_0" FriendlyName="OriginalFile Name test" FileName="rundll32.exe" />
33      <Deny ID="ID_DENY_D_1_0_0" FriendlyName="VBA DLLs" FilePath="%OSDRIVE%\Program Files\Microsoft Office\root\vfs\ProgramFilesCommonX86\Microsoft Shared\VBA\*" />
34      <Deny ID="ID_DENY_D_1_0_1" FriendlyName="VBA DLLs" FilePath="%OSDRIVE%\Program Files\Microsoft Office\root\vfs\ProgramFilesCommonX64\Microsoft Shared\VBA\*" />
35      <Deny ID="ID_DENY_D_1_0_2" FriendlyName="PowerShell" FilePath="%OSDRIVE%\Windows\assembly\NativeImages*" />
36      <Deny ID="ID_DENY_D_1_0_3" FriendlyName="Squiblydoo" FileName="scrobj.dll" />
37      <Deny ID="ID_DENY_D_1_0_4" FriendlyName="AMSI loaded" FileName="amsi.dll" />
38      <Deny ID="ID_DENY_D_1_0_5" FriendlyName="AMSI loaded" FileName="amsiproxy.dll" />
39      <Deny ID="ID_DENY_D_1_0_6" FriendlyName="XLL loaded" FilePath="*.xll" />
40      <Deny ID="ID_DENY_D_1_0_7" FriendlyName="WLL Loaded" FilePath="*.wll" />
```



Converting the XML

The easiest way is through the WDAC Wizard, which is provided by Microsoft.

It is a pretty nice tool, however there is no error handling.
When it fails, it tell you it succeeded but generates no output ;)



Converting the XML

There also is a great set of PowerShell tools, created by Matt Graeber. This has great error handling and will tell you where you made your mistakes.

```
ConvertFrom-CIPolicy : The 'FilePath' attribute is not declared.
At C:\users\student\downloads\WDACTools\BuildAndDeployPolicies.psml:409 char:9
+ ~~~~~
+ ConvertFrom-CIPolicy -XmlFilePath $BasePolicyBuildPath -Binar ...
+ ~~~~~
+ CategoryInfo          : ParserError: (Microsoft.Secur...CIPolicyCommand:ConvertFromCIPolicyCommand) [ConvertFrom-CIPolicy], CIPolicyException
+ FullyQualifiedErrorId : InvalidPolicyFile,Microsoft.SecureBoot.UserConfig.ConvertFromCIPolicyCommand

PS C:\users\student\Desktop> Invoke-WDACCodeIntegrityPolicyBuild @CodeIntegrityPoliciesArgs
ConvertTo-WDACPolicyRuleValue : Cannot validate argument on parameter 'PolicyOptionStrings'. The argument is null, empty, or an element of the argument c
again.
At C:\users\student\downloads\WDACTools\BuildAndDeployPolicies.psml:365 char:92
+ ... WDACPolicyRuleValue -PolicyOptionStrings $CommonBasePolicyRuleOptions
+ ~~~~~
+ CategoryInfo          : InvalidData: (:) [ConvertTo-WDACPolicyRuleValue], ParameterBindingValidationException
+ FullyQualifiedErrorId : ParameterArgumentValidationError,ConvertTo-WDACPolicyRuleValue

ConvertFrom-CIPolicy : Rule with Duplicate Id ID_DENY_D_1_1_1
At C:\users\student\downloads\WDACTools\BuildAndDeployPolicies.psml:409 char:9
+ ~~~~~
+ ConvertFrom-CIPolicy -XmlFilePath $BasePolicyBuildPath -Binar ...
+ ~~~~~
+ CategoryInfo          : InvalidData: (:) [ConvertFrom-CIPolicy], ParameterBindingValidationException
+ FullyQualifiedErrorId : ParameterArgumentValidationError,ConvertFrom-CIPolicy

PolicyType      : Base
XmlFileInfo     : C:\users\student\downloads\WDACTools\BuildArtifacts\BasePolicy.xml
BinaryFileInfo  : C:\users\student\downloads\WDACTools\BuildArtifacts\{A244370E-44C9-4C06-00FF-F6016E563076}.cip
PolicyID        : {A244370E-44C9-4C06-00FF-F6016E563076}
BasePolicyID    : {A244370E-44C9-4C06-00FF-F6016E563076}
PolicyInfoName  : WDACme
PolicyInfoID    : 06_26_2022
```



Introducing WDACme

A WDAC configuration repository with the sole intention of enriching MDE.

This repository contains a Windows Defender Application Control (WDAC) configuration that contains a list of files I always want to know about.

MDE also logs these events to the DeviceEvents table and exposes them in the ActionType *AppControlCodeIntegrityPolicyAudited*

Contains an XML config file and a generated binary config to load.

Currently contains approximately 110 files or locations

<https://github.com/olafhartong/WDACme>



For example; .NET applications

The great thing about this way of configuring is;
Most DLLs have the same OriginalFileName, so 32 and 64bit versions will be captured by the same configuration entry.

Mscoree.dll does not have one in the PE header, hence the FilePath

```
<Deny ID="ID_DENY_D_1_6_7" FriendlyName=".NET Binaries executed" FileName="clr.dll" />  
<Deny ID="ID_DENY_D_1_6_8" FriendlyName=".NET Binaries executed" FileName="clrjit.dll" />  
<Deny ID="ID_DENY_D_1_6_9" FriendlyName=".NET Binaries executed" FileName="mscoreei.dll" />  
<Deny ID="ID_DENY_D_1_7_0" FriendlyName=".NET Binaries executed" FilePath="*\mscoree.dll" />  
<Deny ID="ID_DENY_D_1_7_1" FriendlyName=".NET Binaries executed" FileName="mscoreeis.dll" />  
<Deny ID="ID_DENY_D_1_7_2" FriendlyName=".NET Binaries executed" FileName="mscorlib.dll" />
```



Unexpected .NET applications

```
19 let knownDotNet=dynamic(["sdiaghost.exe","ngentask.exe","ngen.exe","mscorsvw.exe","collectguestlogs.exe","msiexec.exe","powershell.exe","openhandlecollector.exe","csc.exe"]);
20 let trustedStartFolders=dynamic(["c:\windows\microsoft.net\framework64","c:\windows\microsoft.net\framework","c:\windows\system32","c:\programdata\microsoft\windows defender advanced threat protection\datacollection","c:\windowsazure\gu
21 DeviceEvents
22 | where Timestamp > ago(7d)
23 | where ActionType == "AppControlCodeIntegrityPolicyAudited"
24 | extend ValidatedSigningLevel=extractjson("$.Validated Signing Level",AdditionalFields, typeof(string))
25 | where FileName in~ {"clr.dll","clrjit.dll","mscorlib.dll","mscorlib.ni.dll","mscorlib.ni.dll"}
26 | where not(InitiatingProcessFileName in (knownDotNet) and InitiatingProcessFolderPath has_any (trustedStartFolders))
27 | summarize make_set(InitiatingProcessFileName), make_set(InitiatingProcessVersionInfoOriginalFileName), make_set(InitiatingProcessFolderPath) by FileName,FolderPath
28 | extend FolderPath=trim(@"\\Device\\HarddiskVolume4\\",FolderPath)
29
```

Getting Started **Results**

Export

8 items

Search

0:0.62

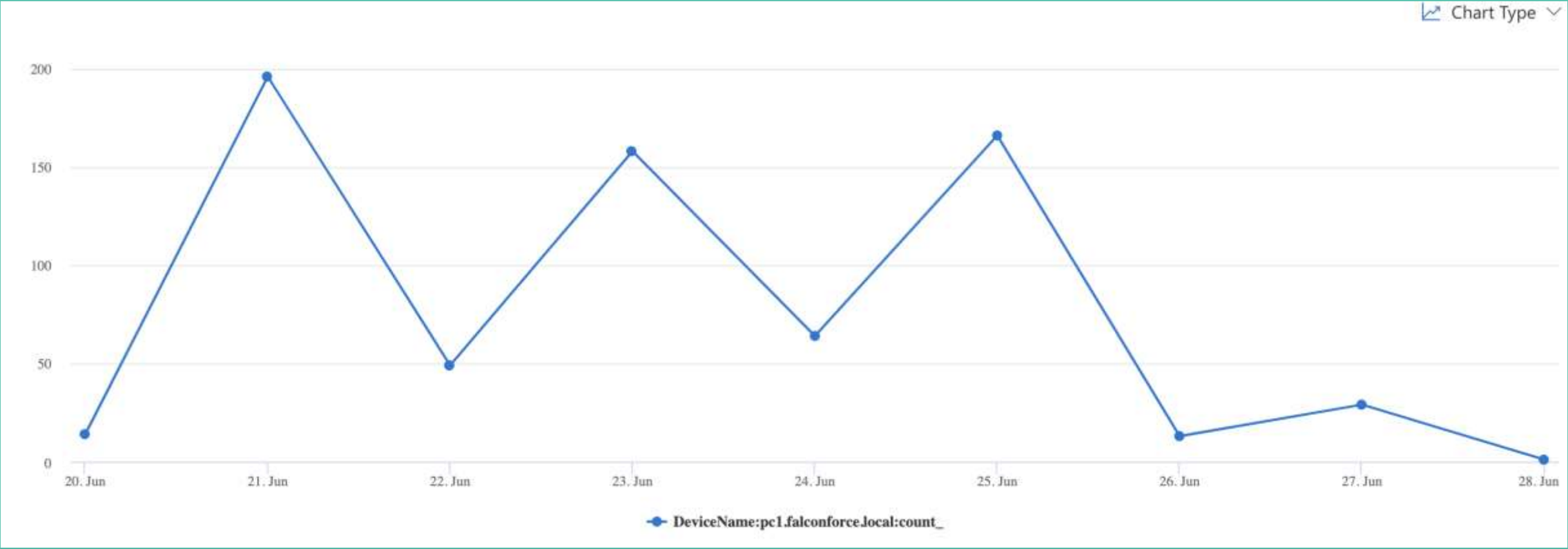
Low

Chart Type Customize column

☐	FileName	FolderPath	set_InitiatingProcessFileName	set_InitiatingProcessVersionInfoOriginalFileName	set_InitiatingProcessFolderPath
☐	mscorlib.dll	Windows\System32	["wdac wizard.exe","gravedigger.exe","gocasey.exe"]	["WDAC Wizard.exe","AutomatedExecution.exe"]	["c:\program files\windowsapps\microsoft.wdac.wdacwizard_1.6.6.1_x64__8wekyb3d8bbwe\w
☐	mscorlib.dll	Windows\Microsoft.NET\Framework64\v4.0.30319	["wdac wizard.exe","gravedigger.exe","gocasey.exe"]	["WDAC Wizard.exe","AutomatedExecution.exe"]	["c:\program files\windowsapps\microsoft.wdac.wdacwizard_1.6.6.1_x64__8wekyb3d8bbwe\w
☐	clr.dll	Windows\Microsoft.NET\Framework64\v4.0.30319	["wdac wizard.exe","gravedigger.exe","gocasey.exe"]	["WDAC Wizard.exe","AutomatedExecution.exe"]	["c:\program files\windowsapps\microsoft.wdac.wdacwizard_1.6.6.1_x64__8wekyb3d8bbwe\w
☐	clrjit.dll	Windows\Microsoft.NET\Framework64\v4.0.30319	["wdac wizard.exe","gravedigger.exe","gocasey.exe"]	["WDAC Wizard.exe","AutomatedExecution.exe"]	["c:\program files\windowsapps\microsoft.wdac.wdacwizard_1.6.6.1_x64__8wekyb3d8bbwe\w
☐	mscorlib.dll	Windows\SysWOW64	["update.exe"]	["Update.exe"]	["c:\users\student\appdata\local\microsoft\teams\update.exe","c:\users\student\appda
☐	mscorlib.dll	Windows\Microsoft.NET\Framework\v4.0.30319	["update.exe"]	["Update.exe"]	["c:\users\student\appdata\local\microsoft\teams\update.exe","c:\users\student\appda
☐	clr.dll	Windows\Microsoft.NET\Framework\v4.0.30319	["update.exe"]	["Update.exe"]	["c:\users\student\appdata\local\microsoft\teams\update.exe","c:\users\student\appda
☐	clrjit.dll	Windows\Microsoft.NET\Framework\v4.0.30319	["update.exe"]	["Update.exe"]	["c:\users\student\appdata\local\microsoft\teams\update.exe","c:\users\student\appda



Average events per day on basic Windows 10 machine



Considerations using WDAC to supplement MDE logging

- The events are capped to 1 event per day per set >
- Not all WDAC fields are loaded in MDE
- This only works for executables and drivers
- The global cap per 24 hours is 500 events
- Monitor policy changes for possible tampering

```
"localCapping": [
  {
    "id": "DGPolicyFailureAuditFirstSeen",
    "expirationPeriodInHours": 24,
    "fields": [
      {
        "fieldName": "File Name"
      },
      {
        "fieldName": "Process Name"
      },
      {
        "fieldName": "Status"
      }
    ],
    "capping": 1
  }
]
```



System vs MDE



Pros and cons per solution

Sysmon

- + Full control over the config and the data you'll get
- + Best applied to augment MDE or in full parallel
- + Rich and unsampled telemetry
- You must maintain it yourself (config, ingestion and detections)
- Only detection, no response

MDE

- + Fully maintained by Microsoft (config and ingestion)
- + Detection and Response capability, custom detections possible in addition
- + Rich set of telemetry, way more than Sysmon
- The configuration is non-configurable
- Telemetry is sampled for most events



Sysmon vs MDE telemetry

Sysmon ID	Sysmon Event Name	MDE Table	ActionType	Notes on MDE
1	Process Creation	DeviceProcessEvents	ProcessCreated	
2	Process Changed a file creation time	n/a	n/a	
3	Network Connection	DeviceNetworkEvents	ConnectionFound, ConnectionSuccess, ConnectionFailed, InboundConnectionAccepted, ListeningConnectionCreated, ConnectionAttempt, ConnectionAcknowledged, ConnectionRequest	Heavily sampled, only 1 st seen event
4	<i>Sysmon Service State Change</i>	-	-	
5	Process Terminated	n/a	n/a	
6	Driver Loaded	DeviceEvents	DriverLoad	
7	Image Loaded	DeviceImageLoadEvents	ImageLoaded	No signer information only hashes Heavily sampled
8	Create Remote Thread	DeviceEvents	CreateRemoteThreadApiCall	Missing info compared to Sysmon: NewThreadId, StartAddress, StartModule, StartFunction
9	<i>Raw File Access Read</i>	n/a	n/a	
10	Process Access	DeviceEvents	ReadProcessMemoryApiCall, WriteToLsassProcessMemory, OpenProcessApiCall	ONLY logged for the lsass.exe process. It does provide TotalBytesCopied on ReadProcessMemoryApiCall. On OpenProcessApiCall it supplies the DesiredAccess in decimal values
11	File Create	DeviceFileEvents	FileCreated	
12	Registry Create and Delete	DeviceRegistryEvents	RegistryKeyCreated, RegistryKeyDeleted, RegistryValueDeleted	Filters are applied
13	Registry Value Set	DeviceRegistryEvents	RegistryValueSet	Filters are applied
14	Registry Key and Value Rename	n/a	n/a	Seems to be there in MDE but often unpopulated
15	File Create Stream Hash	n/a	n/a	
16	<i>Sysmon Config Change</i>	-	-	
17	Pipe Event Created	DeviceEvents	NamedPipeEvent	
18	Pipe Event Connected	n/a	n/a	Only first seen event, connect or create
19	WMI EventFilter activity	n/a	n/a	
20	WMI EventConsumer activity	DeviceEvents	ProcessCreatedUsingWmiQuery	
21	WMI EventConsumerToFilter activity	DeviceEvents	WmiBindEventFilterToConsumer	
22	DNS Query	DeviceEvents	DnsQueryResponse	Response to successful queries
23	FileDelete	DeviceFileEvents	FileDeleted	
24	<i>ClipboardChange</i>	n/a	n/a	
25	Process Tampering	n/a	n/a	
26	FileDeleteDetected	DeviceFileEvents	FileDeleted	No exposed telemetry, it does have alerts for it No file retention option



Sysmon vs MDE – features / telemetry

Sysmon - Unique

Clipboard events saving
Deleted files saving
Preserve deleted PE files
Preserve files for configured processes
Preserve files with configured extensions
Preserve files for configured SIDs

MDE - Unique

DeviceFileEvents	FileRenamed
DeviceFileEvents	FileModified
DeviceLogonEvents	LogonAttempted
DeviceLogonEvents	LogonFailed
DeviceLogonEvents	LogonSuccess
DeviceFileCertificateInfo	-
DeviceInfo	-
DeviceNetworkInfo	-

AntivirusDetection	CredentialsBackup	ProcessPrimaryTokenModified
AntivirusDetectionActionType	DeviceBootAttestationInfo	QueueUserApcRemoteApiCall
AntivirusReport	DnsQueryResponse	ReadProcessMemoryApiCall
AntivirusScanCancelled	DriverLoad	RemoteDesktopConnection
AntivirusScanCompleted	ExploitGuardAcgAudited	RemoteWmiOperation
AntivirusScanFailed	ExploitGuardAcgEnforced	SafeDocFileScan
AppControlCodeIntegritySigningInformation	ExploitGuardChildProcessAudited	ScheduledTaskCreated
AppControlExecutableBlocked	ExploitGuardChildProcessBlocked	ScheduledTaskDeleted
AppControlScriptBlocked	ExploitGuardEafViolationBlocked	ScheduledTaskUpdated
AsrAdobeReaderChildProcessBlocked	ExploitGuardLowIntegrityImageAudited	ScreenshotTaken
AsrExecutableEmailContentBlocked	ExploitGuardLowIntegrityImageBlocked	ScriptContent
AsrExecutableOfficeContentAudited	ExploitGuardNetworkProtectionAudited	SecurityGroupCreated
AsrExecutableOfficeContentBlocked	ExploitGuardNonMicrosoftSignedAudited	SecurityGroupDeleted
AsrLsassCredentialTheftAudited	ExploitGuardNonMicrosoftSignedBlocked	SecurityLogCleared
AsrLsassCredentialTheftBlocked	ExploitGuardSharedBinaryAudited	SensitiveFileRead
AsrObfuscatedScriptAudited	ExploitGuardSharedBinaryBlocked	ServiceInstalled
AsrOfficeChildProcessAudited	ExploitGuardWin32SystemCallBlocked	SetThreadContextRemoteApiCall
AsrOfficeChildProcessBlocked	FirewallInboundConnectionBlocked	ShellLinkCreateFileEvent
AsrOfficeCommAppChildProcessAudited	FirewallInboundConnectionToAppBlocked	SmartScreenAppWarning
AsrOfficeCommAppChildProcessBlocked	FirewallOutboundConnectionBlocked	SmartScreenExploitWarning
AsrOfficeMacroWin32ApiCallsAudited	GetAsyncKeyStateApiCall	SmartScreenUrlWarning
AsrOfficeMacroWin32ApiCallsBlocked	GetClipboardData	SmartScreenUserOverride
AsrOfficeProcessInjectionAudited	LdapSearch	UntrustedWifiConnection
AsrOfficeProcessInjectionBlocked	MemoryRemoteProtect	UsbDriveDriveLetterChanged
AsrPsexecWmiChildProcessAudited	NamedPipeEvent	UsbDriveMounted
AsrRansomwareBlocked	NtAllocateVirtualMemoryApiCall	UsbDriveUnmounted
AsrUntrustedExecutableAudited	NtAllocateVirtualMemoryRemoteApiCall	UserAccountAddedToLocalGroup
AsrUntrustedUsbProcessAudited	NtMapViewOfSectionRemoteApiCall	UserAccountCreated
AsrUntrustedUsbProcessBlocked	NtProtectVirtualMemoryApiCall	UserAccountDeleted
AuditPolicyModification	OpenProcessApiCall	UserAccountModified
BluetoothPolicyTriggered	OtherAlertRelatedActivity	UserAccountRemovedFromLocalGroup
BrowserLaunchedToOpenUrl	PnpDeviceAllowed	WmiBindEventFilterToConsumer
ControlFlowGuardViolation	PnpDeviceBlocked	WriteToLsassProcessMemory
ControlledFolderAccessViolationAudited	PnpDeviceConnected	
ControlledFolderAccessViolationBlocked	PowerShellCommand	
CreateRemoteThreadApiCall	ProcessCreatedUsingWmiQuery	

.... 181 in total



MDE telemetry potential mapping to MITRE ATT&CK

Initial Access 9 techniques	Execution 10 techniques	Persistence 18 techniques	Privilege Escalation 13 techniques	Defense Evasion 32 techniques	Credential Access 15 techniques	Discovery 25 techniques	Lateral Movement 9 techniques	Collection 15 techniques	Command and Control 16 techniques	Exfiltration 8 techniques	Impact 13 techniques
Drive-by Compromise	Command and Scripting Interpreter (5/5)	Account Manipulation (1/3)	Abuse Elevation Control Mechanism (1/1)	Abuse Elevation Control Mechanism (1/1)	Brute Force (4/4)	Account Discovery (3/4)	Exploitation of Remote Services	Archive Collected Data (3/3)	Application Layer Protocol (0/4)	Automated Exfiltration (0/0)	Account Access Removal
Exploit Public-Facing Application	Exploitation for Client Execution	BITS Jobs	Access Token Manipulation (5/5)	Access Token Manipulation (5/5)	Credentials from Password Stores (3/3)	Application Window Discovery	Internal Spearphishing	Audio Capture	Communication Through Removable Media	Data Transfer Size Limits	Data Destruction
External Remote Services	Inter-Process Communication (2/2)	Boot or Logon Autostart Execution (10/10)	Boot or Logon Autostart Execution (10/10)	BITS Jobs	Exploitation for Credential Access	Browser Bookmark Discovery	Lateral Tool Transfer	Automated Collection	Data Encoding (0/2)	Exfiltration Over Alternative Protocol (3/3)	Data Encrypted for Impact
Hardware Additions	Native API	Boot or Logon Initialization Scripts (2/2)	Boot or Logon Initialization Scripts (10/10)	Deobfuscate/Decode Files or Information	Forced Authentication	Cloud Service Dashboard	Remote Service Session Hijacking (1/1)	Clipboard Data	Data Obfuscation (0/3)	Exfiltration Over C2 Channel	Data Manipulation (3/3)
Phishing (0/3)	Scheduled Task/Job (2/2)	Browser Extensions	Boot or Logon Initialization Scripts (2/2)	Direct Volume Access	Forge Web Credentials (2/2)	Cloud Service Discovery	Remote Services (5/5)	Data from Information Repositories (1/1)	Dynamic Resolution (2/3)	Exfiltration Over Other Network Medium (1/1)	Defacement (2/2)
Replication Through Removable Media	Shared Modules	Compromise Client Software Binary	Create or Modify System Process (1/1)	Domain Policy Modification (2/2)	Input Capture (4/4)	Domain Trust Discovery	Replication Through Removable Media	Data from Local System	Encrypted Channel (0/2)	Exfiltration Over Physical Medium (1/1)	Disk Wipe (2/2)
Supply Chain Compromise (0/3)	Software Deployment Tools	Create Account (2/3)	Domain Policy Modification (2/2)	Execution Guardrails (1/1)	Man-in-the-Middle (1/2)	File and Directory Discovery	Software Deployment Tools	Data from Network Shared Drive	Fallback Channels	Exfiltration Over Web Service (2/2)	Endpoint Denial of Service (0/4)
Trusted Relationship	System Services (1/1)	Create or Modify System Process (1/1)	Escape to Host	File and Directory Permissions Modification (1/1)	Modify Authentication Process (2/2)	Network Service Scanning	Taint Shared Content	Data from Removable Media	Ingress Tool Transfer	Scheduled Transfer	Firmware Corruption
Valid Accounts (3/4)	User Execution (2/2)	Event Triggered Execution (11/11)	Event Triggered Execution (11/11)	Hide Artifacts (6/6)	Network Sniffing	Network Share Discovery	Use Alternate Authentication Material (2/4)	Data Staged (2/2)	Multi-Stage Channels		Inhibit System Recovery
	Windows Management Instrumentation	External Remote Services	Exploitation for Privilege Escalation	Hijack Execution Flow (9/9)	OS Credential Dumping (5/6)	Network Sniffing		Email Collection (2/3)	Non-Application Layer Protocol		Network Denial of Service (0/2)
		Hijack Execution Flow (9/9)	Hijack Execution Flow (9/9)	Impair Defenses (5/5)	Steal Application Access Token	Permission Groups Discovery (2/3)		Input Capture (4/4)	Non-Standard Port		Resource Hijacking
		Modify Authentication Process (2/2)	Process Injection (8/8)	Indicator Removal on Host (5/5)	Steal or Forge Kerberos Tickets (0/4)	Process Discovery		Man in the Browser	Protocol Tunneling		Service Stop
		Office Application Startup (6/6)	Scheduled Task/Job (2/2)	Indirect Command Execution	Steal Web Session Cookie	Query Registry		Man-in-the-Middle (1/2)	Proxy (3/4)		System Shutdown/Reboot
		Pre-OS Boot (1/3)	Valid Accounts (3/4)	Process Injection (T1055)	Two-Factor Authentication	Remote System Discovery		Screen Capture	Remote Access Software		
		Scheduled Task/Job (2/2)		Score: 80	DeviceFileEvents_FileModified	Software Discovery (1/1)		Video Capture	Traffic Signaling (1/1)		
		Server Software Component (2/3)		Aggregate Score (average): 92.22	DeviceFileEvents_FileRenamed	System Information Discovery			Web Service (2/3)		
		Traffic Signaling (1/1)		file modification:	DeviceImageLoadEvents_ImageLoaded	System Location Discovery					
		Valid Accounts (3/4)		module load:	DeviceEvents_OpenProcessApiCall	System Network Configuration Discovery (1/1)					
				Obfuscated Files or process access:	DeviceEvents_CreateRemoteThreadApiCall	System Network Connections Discovery					
				OS api execution:	Pre-OS Boot (1/3)	System Owner/User Discovery					
				Process Injection (8/8)	Rogue Domain Controller						
				Rootkit							
				Signed Binary Proxy Execution (11/11)							

Sysmon telemetry potential mapping to MITRE ATT&CK

Initial Access 9 techniques	Execution 10 techniques	Persistence 18 techniques	Privilege Escalation 13 techniques	Defense Evasion 32 techniques	Credential Access 15 techniques	Discovery 25 techniques	Lateral Movement 9 techniques	Collection 15 techniques	Command and Control 16 techniques	Exfiltration 8 techniques	Impact 13 techniques
Drive-by Compromise	Command and Scripting Interpreter (5/5)	Account Manipulation (0/3)	Abuse Elevation Control Mechanism (1/1)	Abuse Elevation Control Mechanism (1/1)	Brute Force (0/4)	Account Discovery (3/4)	Exploitation of Remote Services	Archive Collected Data (3/3)	Application Layer Protocol (0/4)	Automated Exfiltration (0/0)	Account Access Removal
Exploit Public-Facing Application	Exploitation for Client Execution	BITS Jobs	Access Token Manipulation (5/5)	Access Token Manipulation (5/5)	Credentials from Password Stores (3/3)	Application Window Discovery	Internal Spearphishing	Audio Capture	Communication Through Removable Media	Data Transfer Size Limits	Data Destruction
External Remote Services	Inter-Process Communication (2/2)	Boot or Logon Autostart Execution (10/10)	Boot or Logon Autostart Execution (10/10)	BITS Jobs	Exploitation for Credential Access	Browser Bookmark Discovery	Lateral Tool Transfer	Automated Collection	Data Encoding (0/2)	Exfiltration Over Alternative Protocol (3/3)	Data Encrypted for Impact
Hardware Additions	Native API	Boot or Logon Initialization Scripts (2/2)	Boot or Logon Initialization Scripts (10/10)	Deobfuscate/Decode Files or Information	Forced Authentication	Cloud Service Dashboard	Remote Service Session Hijacking (1/1)	Clipboard Data	Data Obfuscation (0/3)	Exfiltration Over C2 Channel	Data Manipulation (3/3)
Phishing (0/3)	Scheduled Task/Job (2/2)	Browser Extensions	Boot or Logon Initialization Scripts (2/2)	Direct Volume Access	Forge Web Credentials (0/2)	Cloud Service Discovery	Replication Through Removable Media	Data from Information Repositories (0/1)	Dynamic Resolution (2/3)	Exfiltration Over Other Network Medium (1/1)	Defacement (2/2)
Replication Through Removable Media	Shared Modules	Compromise Client Software Binary	Create or Modify System Process (1/1)	Domain Policy Modification (2/2)	Input Capture (4/4)	Domain Trust Discovery	Software Deployment Tools	Data from Local System	Encrypted Channel (0/2)	Exfiltration Over Physical Medium (1/1)	Disk Wipe (2/2)
Supply Chain Compromise (0/3)	Software Deployment Tools	Create Account (2/3)	Domain Policy Modification (2/2)	Exploitation for Defense Evasion	Man-in-the-Middle (1/2)	File and Directory Discovery	Taint Shared Content	Data from Network Shared Drive	Fallback Channels	Scheduled Transfer	Endpoint Denial of Service (0/4)
Trusted Relationship	System Services (1/1)	Create or Modify System Process (1/1)	Escape to Host	File and Directory Permissions Modification (1/1)	Modify Authentication Process (2/2)	Network Service Scanning	Use Alternate Authentication Material (0/4)	Data from Removable Media	Ingress Tool Transfer	Exfiltration Over Web Service (2/2)	Firmware Corruption
Valid Accounts (0/4)	User Execution (2/2)	Event Triggered Execution (11/11)	Event Triggered Execution (11/11)	Hide Artifacts (5/6)	Network Sniffing	Network Share Discovery		Data Staged (2/2)	Multi-Stage Channels		Inhibit System Recovery
	Windows Management Instrumentation	External Remote Services	Exploitation for Privilege Escalation	Hijack Execution Flow (9/9)	OS Credential Dumping (5/6)	Network Sniffing		Email Collection (2/3)	Non-Application Layer Protocol		Network Denial of Service (0/2)
		Hijack Execution Flow (9/9)	Hijack Execution Flow (9/9)	Impair Defenses (5/5)	Steal Application Access Token	Password Policy Discovery		Input Capture (4/4)	Non-Standard Port		Resource Hijacking
		Modify Authentication Process (2/2)	Process Injection (8/8)	Process Injection (T1055)	Steal or Forge Kerberos Tickets (0/4)	Permission Groups Discovery (2/3)		Man in the Browser	Protocol Tunneling		Service Stop
		Office Application Startup (6/6)	Scheduled Task/Job (2/2)	Score: 80		Process Discovery		Man-in-the-Middle (1/2)	Proxy (3/4)		System Shutdown/Reboot
		Pre-OS Boot (1/3)	Valid Accounts (0/4)	Aggregate Score (average): 92.22		Query Registry		Screen Capture	Remote Access Software		
		Scheduled Task/Job (2/2)		file modification:		Remote System Discovery		Video Capture	Traffic Signaling (1/1)		
		Server Software Component (2/3)		Masquerading (2/5)		Software Discovery (1/1)			Web Service (2/3)		
		Traffic Signaling (1/1)		Modify Authentication Process		System Information Discovery					
		Valid Accounts (0/4)		Module load:		System Location Discovery					
				Modify Registry process access:		System Network Configuration Discovery (1/1)					
				Obfuscated Files or Information		System Network Connections Discovery					
				OS api execution:		System Owner/User Discovery					
				Pre-OS Boot (1/3)		System Service Discovery					
				Rogue Domain Controller							
				Rootkit							
				Signed Binary Proxy Execution (11/11)							

Wrapping up

- Know your tools, understand their strengths and weaknesses
- Understand what your tools are detecting and HOW they are detecting it
- Continuously reassess this to see what is new or improved
- Augment the weak or blind spots with additional tools
- This talk is about 1 vendors product. Trust me, most EDRs have similar issues.





Thank you! Questions ?



olaf@falconforce.nl



<https://falconforce.nl>



[@olafhartong](https://twitter.com/olafhartong)
[@falconforceteam](https://twitter.com/falconforceteam)



<https://linkedin.com/in/olafhartong>

Time

It's not always on your side

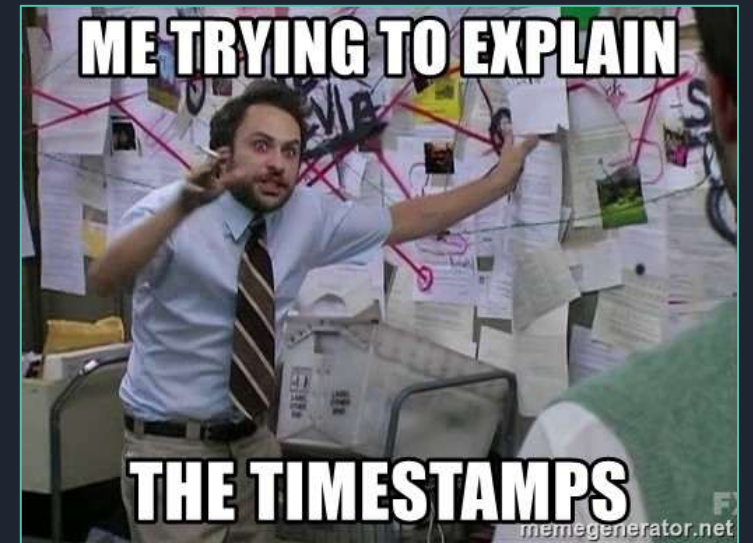


Time

The generic running joke amongst a lot of defenders is that if it didn't get recorded in UTC it didn't happen.

However, time zones are not our only worry.

Time ingestion, and its delay(s), also play a crucial factor in building solid detections.



Time - Expected behavior

Expected behavior: event happens at a certain time, gets shipped to the cloud log store and will be visible within a few seconds.



“Should happen” behavior

Event happens at a certain time, is stored in a buffer locally and then at periodic intervals they're shipped in batches to the cloud log store and are visible after a few minutes.



Actual behavior

Event happens at a certain time, is stored in a buffer locally and then at periodic intervals they're shipped in batches to the cloud log store and can take a while to become visible.



Ingestion delay in a lab environment

[Run query](#) [Save](#) [Share link](#) Last 30 days [Create detection rule](#)

Query

```
1 // Grab all tables and add the Table field with the value name of the sourceTable
2 union withsource=Table *
3 // Filter all tables that are most commonly used on MDE events
4 | where Table startswith "Device" and Table endswith "Events"
5 // Add the ingestion_time to a field so we can use it in the diff
6 | extend IngestTime = ingestion_time()
7 // use the datetime_diff function to calculate the time difference in minutes between the ingestion_time and the generated time on the host
8 | extend Diff = datetime_diff("minute", IngestTime, Timestamp)
9 // create a table with the fields we care about
10 | project Table, Timestamp, IngestTime, Diff
11 // summarize all events and calculate the average time difference and a set of percentiles to determine our coverage in a certain delay
12 | summarize avg(Diff), percentiles(Diff, 75, 95, 96, 97, 98, 99) by Table
13 | sort by percentile_Diff_99 asc
```

Getting Started **Results**

[Export](#) 7 items ⌚ 0:0.141 Low [Chart Type](#) [Customize columns](#)

Table	avg_Diff	percentile_Diff_75	percentile_Diff_95	percentile_Diff_96	percentile_Diff_97	percentile_Diff_98	percentile_Diff_99
DeviceFileEvents	16.592049044561183	8	12	17	19	21	49
DeviceProcessEvents	27.460596428595128	7	9	9	11	19	79
DeviceImageLoadEvents	31.301439405389548	7	10	12	18	24	327
DeviceEvents	27.44526769013953	7	9	10	11	20	487
DeviceLogonEvents	42.918243754731265	8	10	15	21	61	1312
DeviceNetworkEvents	41.12756465667134	7	9	10	14	42	1568
DeviceRegistryEvents	57.46462792902025	7	10	11	24	523	2373



Ingestion delay in a real environment (~65k devices)

[Run query](#) [Save](#) [Share link](#) Last 30 days [Create detection rule](#)

Query

```
2 union withsource=Table *
3 // Filter all tables that are most commonly used on MDE events
4 | where Table startswith "Device" and Table endswith "Events"
5 // Add the ingestion_time to a field so we can use it in the diff
6 | extend IngestTime = ingestion_time()
7 // use the datetime_diff function to calculate the time difference in minutes between the ingestion_time and the generated time on the host
8 | extend Diff = datetime_diff("minute", IngestTime, Timestamp)
9 // create a table with the fields we care about
10 | project Table, Timestamp, IngestTime, Diff
11 // summarize all events and calculate the average time difference and a set of percentiles to determine our coverage in a certain delay
12 | summarize avg(Diff), percentiles(Diff, 75, 95, 96, 97, 98, 99) by Table
13 | sort by percentile_Diff_99 asc
```

Getting Started

Results

[Export](#)

7 items

0:9.235 Medium [Chart Type](#) [Customize columns](#)

Table	avg_Diff	percentile_Diff_75	percentile_Diff_95	percentile_Diff_96	percentile_Diff_97	percentile_Diff_98	percentile_Diff_99
DeviceFileEvents	36.30774791151264	8	25	37	57	153	811
DeviceNetworkEvents	41.54270855615515	8	32	50	76	241	937
DeviceImageLoadEvents	46.386571265467815	9	38	57	119	412	1066
DeviceEvents	44.6157926475188	6	39	57	123	432	1073
DeviceRegistryEvents	45.11525047922749	6	32	50	86	316	1147
DeviceProcessEvents	52.51039567860166	10	45	67	165	547	1148
DeviceLogonEvents	54.5295171743802	4	47	102	439	836	1159



Why does this matter ? - Scheduled searches

- TimeGenerated search, running each hour
| where TimeGenerated > ago(1h)

Generated:
13:37:00 UTC



Run time:
13:00:00 UTC
Scope:
12:00-13:00



Run time:
14:00:00 UTC
Scope:
13:00-14:00



Ingested:
14:47:10 UTC



Run time:
15:00:00 UTC
Scope:
14:00-15:00

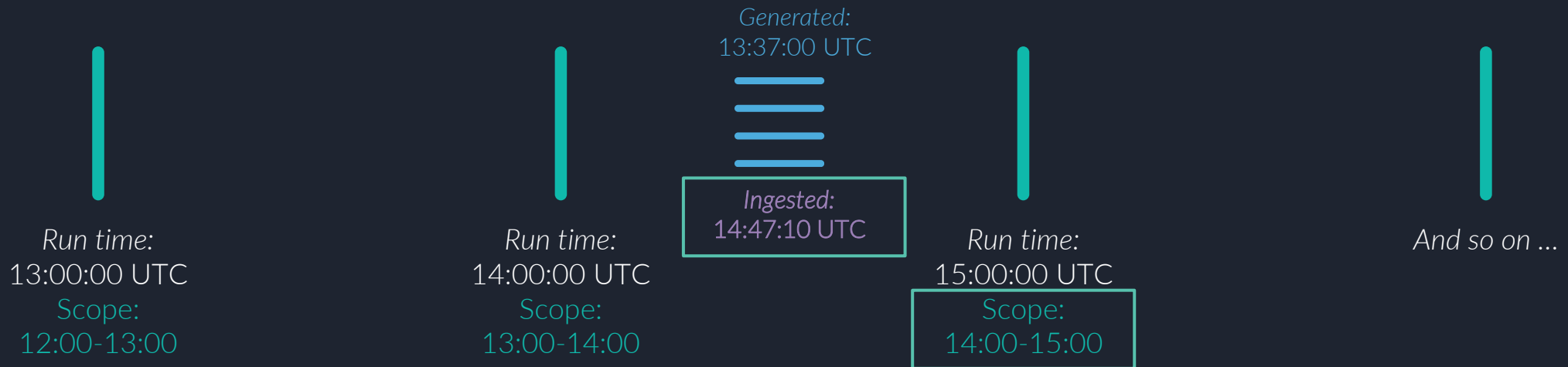


And so on ...

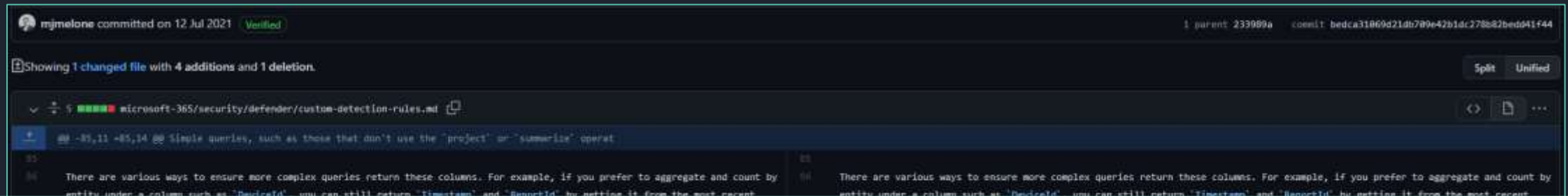


Time – Better way of working

- Use `ingestion_time()` in stead of `TimeGenerated/Timestamp`
| where `ingestion_time > ago(1h)`



Time – Microsoft agrees (partially)



>[!IMPORTANT]

>When performing time-based filtering in a custom detection be sure to use `ingestion_time()` instead of `Timestamp` to avoid data loss. `Timestamp` represents the time that an event occurred, whereas `ingestion_time()` represents when the record was written to the Advanced Hunting table. This can cause an issue if the event is cached on the endpoint due to lack of Internet connectivity and uploaded later. Also, note that queries used in custom detections are pre-filtered depending on the custom detection frequency.

```
90 93 ```kusto
91 94 DeviceEvents
92 - | where Timestamp > ago(1d)
95 + | where ingestion_time() > ago(1d)
93 96 | where ActionType == "AntivirusDetection"
94 97 | summarize (Timestamp, ReportId)=arg_max(Timestamp, ReportId), count() by DeviceId
95 98 | where count_ > 5
```



Custom detection considerations

If you schedule an alert for a specific time, the portal prepends some (hidden) logic to it, setting the lookback time and querying for that timeframe.

Query Frequency → Lookback time of the **generated time** (Timestamp)

- 1 hour – Lookback time is set to 2 hours
- 3 hours – Lookback time is set to 6 hours
- 12 hours – Lookback time is set to 24 hours
- 24 hours – Lookback time is set to 30 days



Ingestion delay rule coverage, rules scheduled each hour

[Run query](#) [Save](#) [Share link](#) Last 30 days [Create detection rule](#)

Query

```
2 union withsource=Table *
3 // Filter all tables that are most commonly used on MDE events
4 | where Table startswith "Device" and Table endswith "Events"
5 // Add the ingestion_time to a field so we can use it in the diff
6 | extend IngestTime = ingestion_time()
7 // use the datetime_diff function to calculate the time difference in minutes between the ingestion_time and the generated time on the host
8 | extend Diff = datetime_diff("minute", IngestTime, Timestamp)
9 // create a table with the fields we care about
10 | project Table, Timestamp, IngestTime, Diff
11 // summarize all events and calculate the average time difference and a set of percentiles to determine our coverage in a certain delay
12 | summarize avg(Diff), percentiles(Diff, 75, 95, 96, 97, 98, 99) by Table
13 | sort by percentile_Diff_99 asc
```

Getting Started

Results

[Export](#)

7 items

0:9.235 Medium [Chart Type](#) [Customize columns](#)

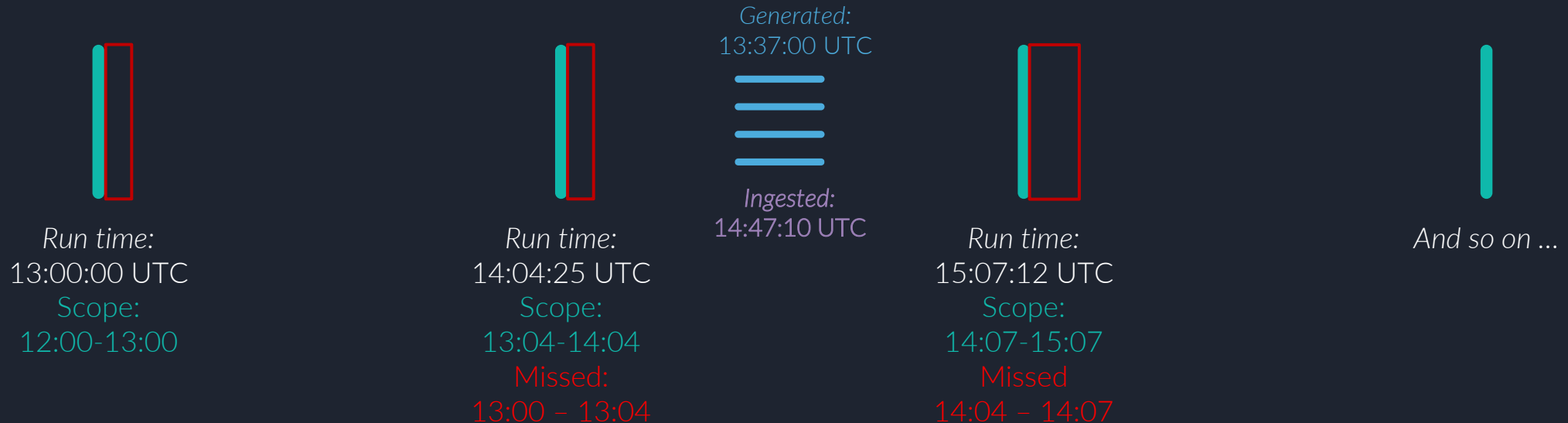
Table	avg_Diff	percentile_Diff_75	percentile_Diff_95	percentile_Diff_96	percentile_Diff_97	percentile_Diff_98	percentile_Diff_99
DeviceFileEvents	36.30774791151264	8	25	37	57	153	811
DeviceNetworkEvents	41.54270855615515	8	32	50	76	241	937
DeviceImageLoadEvents	46.386571265467815	9	38	57	119	412	1066
DeviceEvents	44.6157926475188	6	39	57	123	432	1073
DeviceRegistryEvents	45.11525047922749	6	32	50	86	316	1147
DeviceProcessEvents	52.51039567860166	10	45	67	165	547	1148
DeviceLogonEvents	54.5295171743802	4	47	102	439	836	1159



Scheduler issues

The MDE scheduler isn't consistent.

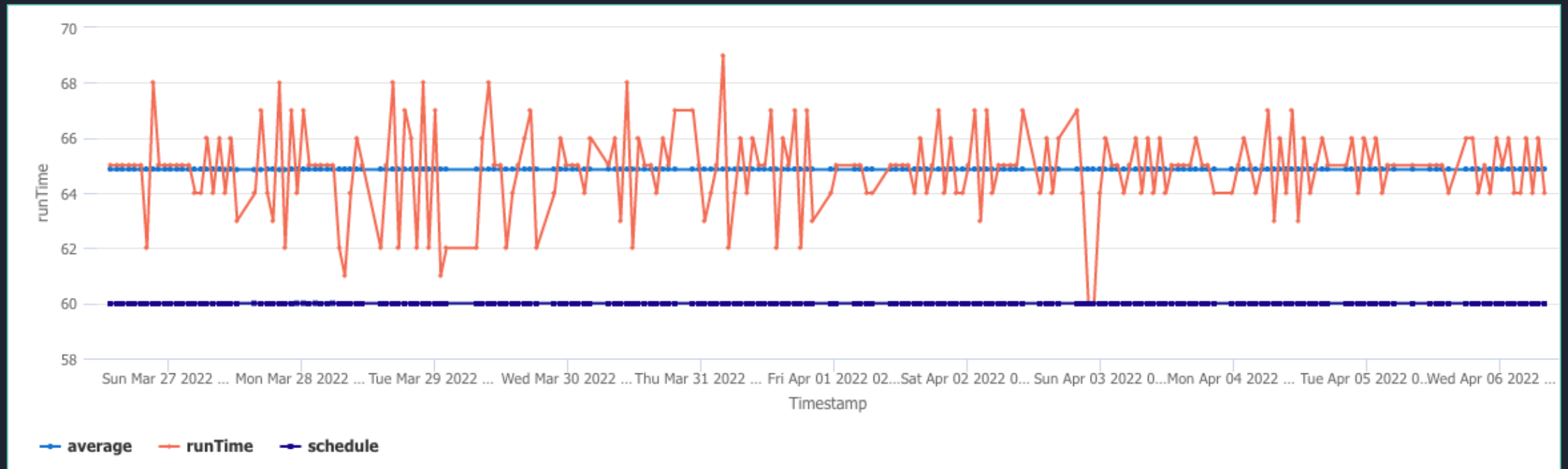
When set to run each hour it isn't running every 60 minutes.



Scheduler timing

Average run time is 64.85 minutes, not the scheduled 60 minutes.

This means we have a blind spot in between searches if we scope for 1 hour.



So, we just process all data?

Well, no, certainly not in all cases.

Processing data is not limitless, and at scale this takes quite a bit of resources. Since this service is based on **shared resource clusters** there are quotas.

 This query used **39%** of your organization's allocated resources for the current 15 minutes. To read our best practices on how to run more efficient queries, click [here](#).

 This query used **47%** of your organization's allocated resources for the current 15 minutes. To read our best practices on how to run more efficient queries, click [here](#).

 This query used **100%** of your organization's allocated resources for the current 15 minutes. To read our best practices on how to run more efficient queries, click [here](#).

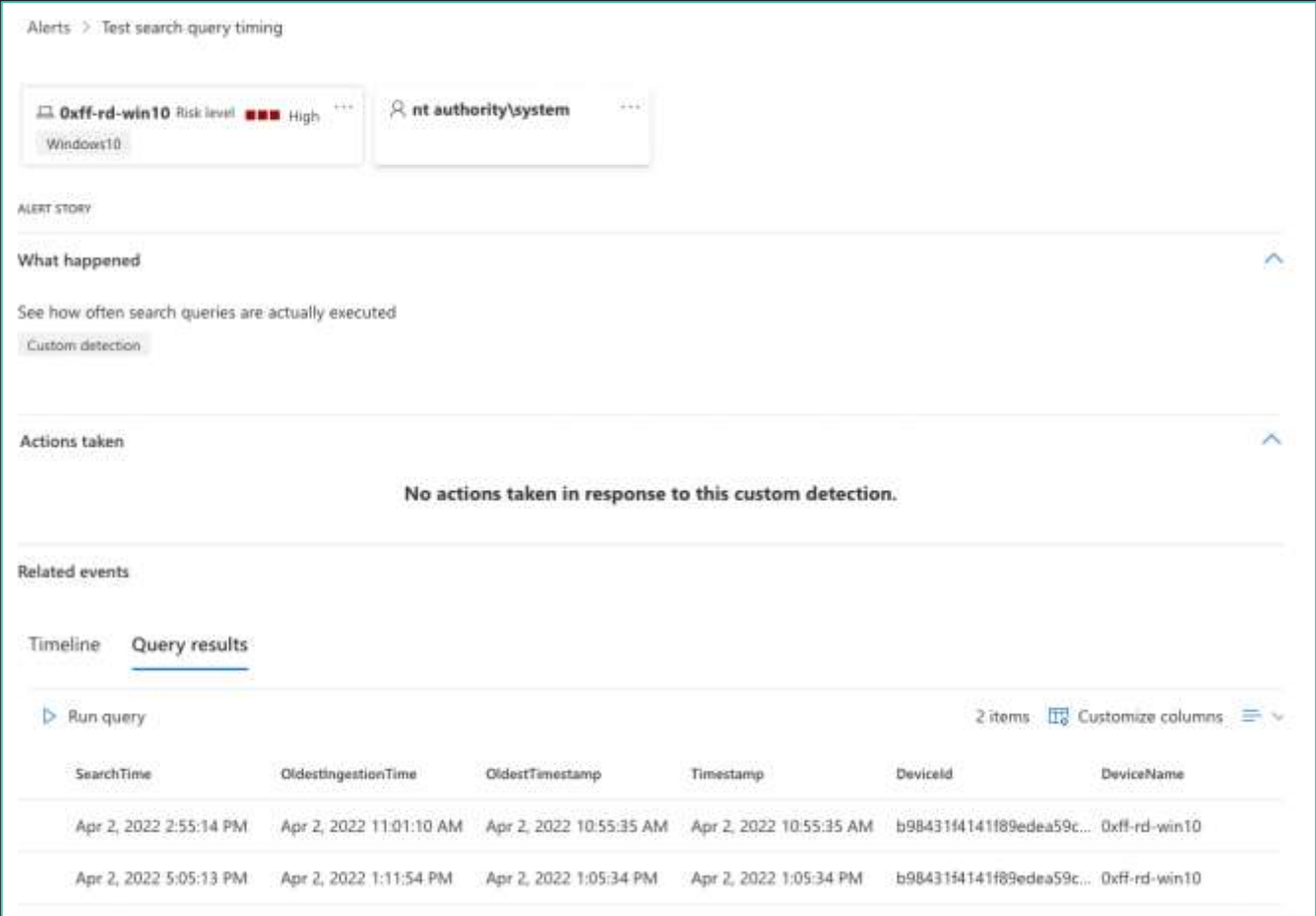


Another reason for scoping your processed timeframe

Looking at, for instance, 2 hours of data every hour will likely cause duplicate alerts.

The MDE portal can *theoretically* take care of duplicate alerts.

This **ONLY** works when they are not closed yet. So we need to add some additional Kusto to take care of this.



Alerts > Test search query timing

Oxfl-rd-win10 Risk level ■■■ High *** **nt authority\system** ***

Windows10

ALERT STORY

What happened

See how often search queries are actually executed

Custom detection

Actions taken

No actions taken in response to this custom detection.

Related events

Timeline **Query results**

Run query 2 items [Customize columns](#)

SearchTime	OldestIngestionTime	OldestTimestamp	Timestamp	DeviceId	DeviceName
Apr 2, 2022 2:55:14 PM	Apr 2, 2022 11:01:10 AM	Apr 2, 2022 10:55:35 AM	Apr 2, 2022 10:55:35 AM	b98431f4141f89ede59c...	Oxfl-rd-win10
Apr 2, 2022 5:05:13 PM	Apr 2, 2022 1:11:54 PM	Apr 2, 2022 1:05:34 PM	Apr 2, 2022 1:05:34 PM	b98431f4141f89ede59c...	Oxfl-rd-win10



Deduplicating results

We can achieve this by using the `AlertInfo` and `AlertEvidence` tables and remove duplicate results based on the same Entity and Timestamp.

Additional details are sadly not all exposed in the KQL tables, so in case of multiple results only the first is queryable.

Deduplication therefore has to happen on the impacted entity;

- DeviceName (if the entity is a machine)
- AccountName (if the entity is a user)
- AccountUpn (if the entity is a mailbox)



Deduplicating results – Rule logic

```
// use 2x the timeframe you're actually interested in.
// make sure this is possible in the lookback period set by the rule engine.
let timeframe = 2*1h;
let RuleId = "[CUSTOM RuleId, like 0042]";
// This has been added to keep the logic below identical for all rules, we
template everything in code
let DedupField = "DeviceName";

// [PLACE DETECTION RULE HERE]
// [make sure to use
// | where ingestion_time() >= ago(timeframe)

// [Paste at the end of a rule]
// Begin de-duplication logic.
// We require a unique value so we don't accidentally introduce false negatives if the value of the DedupField column is empty or
// doesn't exist.
| extend DedupEntity = column_ifexists(DedupField, strcat("N/A"))
// remove the results of the query above when there is an alert for this DeviceName, with the same RuleId and in the same Timeframe
| join kind=leftanti (
AlertInfo
| where Timestamp >= ago(timeframe)
| where Title contains RuleId
| join kind=leftouter AlertEvidence on AlertId
// only search for alerts with an impacted entity and which has been raised by a Custom Detection
| where EvidenceRole == "Impacted" and DetectionSource == "Custom detection"
//the default value if the column doesn't exist is on purpose different than above to prevent accidental matches.
| extend DedupEntity = column_ifexists(DedupField, "")
// remove results where the DedupEntity is unpopulated
| where not(isempty(DedupEntity))
| project DedupEntity
) on DedupEntity
// End de-duplication logic.
```



Audit Success	3/14/2022 6:27:23 PM	Microsoft Windows security auditing.	4688	Process Creation
Audit Success	10/26/2021 12:00:00 AM	Microsoft Windows security auditing.	4688	Process Creation
Audit Success	3/14/2022 6:27:18 PM	Microsoft Windows security auditing.	4688	Process Creation
Audit Success	3/14/2022 6:32:16 PM	Microsoft Windows security auditing.	4688	Process Creation

Event 4688, Microsoft Windows security auditing.

General Details

A new process has been created.

Creator Subject:

Security ID: PC1\student
Account Name: student
Account Domain: PC1
Logon ID: 0x35EBC2

Target Subject:

Security ID: NULL SID
Account Name: -
Account Domain: -
Logon ID: 0x0

Process Information:

New Process ID: 0x3e8
New Process Name: C:\Windows\System32\notepad.exe
Token Elevation Type: %%1936
Mandatory Label: Mandatory Label\High Mandatory Level
Creator Process ID: 0x20c4
Creator Process Name: C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
Process Command Line: "C:\Windows\system32\notepad.exe"

Token Elevation Type indicates the type of token that was assigned to the new process in accordance with User Account Control policy.

Type 1 is a full token with no privileges removed or groups disabled. A full token is only used if User Account Control is disabled or if the user is the built-in Administrator account or a service account.

Type 2 is an elevated token with no privileges removed or groups disabled. An elevated token is used when User Account Control is enabled and the user chooses to start the program using Run as administrator. An elevated token is also used when an application is configured to always require administrative privilege or to always require maximum privilege, and the user is a member of the Administrators group.

Type 3 is a limited token with administrative privileges removed and administrative groups disabled. The limited token is used when User Account Control is enabled, the application does not require administrative privilege, and the user does not choose to start the program using Run as administrator.

Log Name: Security

Source: Microsoft Windows security Logged: 10/26/2021 12:00:00 AM

Event ID: 4688 Task Category: Process Creation

Level: Information Keywords: Audit Success

User: N/A Computer: PC1.FalconForce.local

OpCode: Info

More Information: [Event Log Online Help](#)

```
12
13 DeviceProcessEvents | where ingestion_time() > ago(1h)
14 | where ProcessCommandLine contains "notepad"
15 | extend IngestionTime=ingestion_time()
16 | project Timestamp,ProcessCreationTime,IngestionTime
```

Getting Started Results

↓ Export

Timestamp (UTC)	ProcessCreationTime	IngestionTime
Mar 14, 2022 6:22:49 PM	Oct 26, 1985 12:00:00 AM	Mar 14, 2022 6:27:07 PM
Mar 14, 2022 6:27:19 PM	Oct 26, 2021 12:00:00 AM	Mar 14, 2022 6:32:16 PM



Summarizing time issues

When building custom detections:

- Don't use `/ where Timestamp`, use `| where ingestion_time()`
- Take the ingestion delay into account for your time scoping
- Take the account rule scheduling delays into account
- Make sure to deduplicate alerts when extending the timeframe
- Write rules according to the best practices to keep the performance impact to a minimum



Wrapping up

- Know your tools, understand their strengths and weaknesses
- Understand what your tools are detecting and HOW they are detecting it
- Continuously reassess this to see what is new or improved
- Augment the weak or blind spots with additional tools
- This talk is about 1 vendors product. Trust me, most EDRs have similar issues.





Thank you! Questions ?



olaf@falconforce.nl



<https://falconforce.nl>



[@olafhartong](https://twitter.com/olafhartong)
[@falconforceteam](https://twitter.com/falconforceteam)



<https://linkedin.com/in/olafhartong>