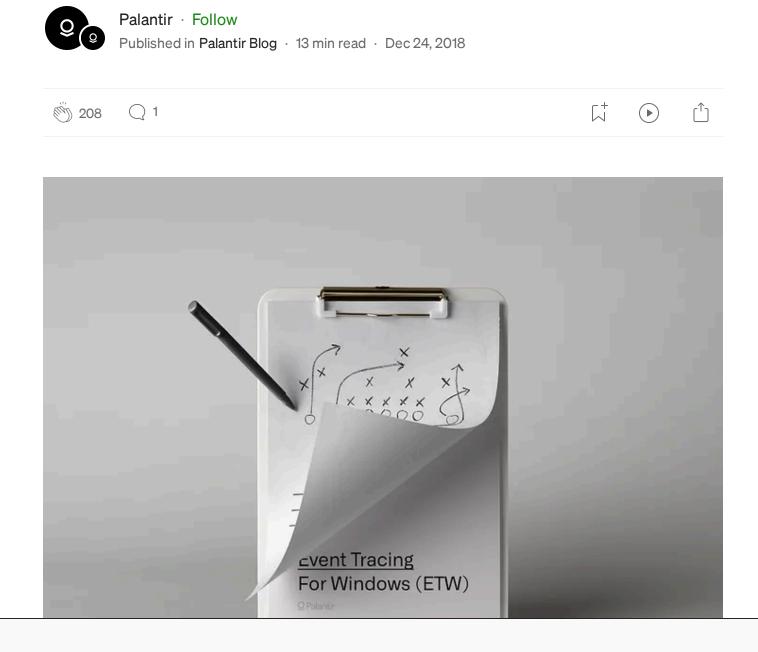
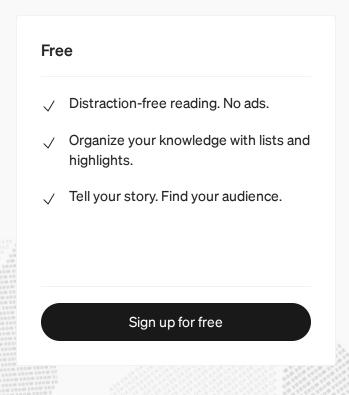
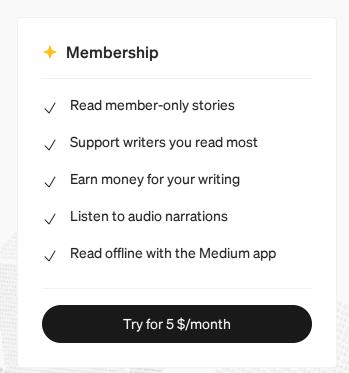


Tampering with Windows Event Tracing: Background, Offense, and Defense



Medium





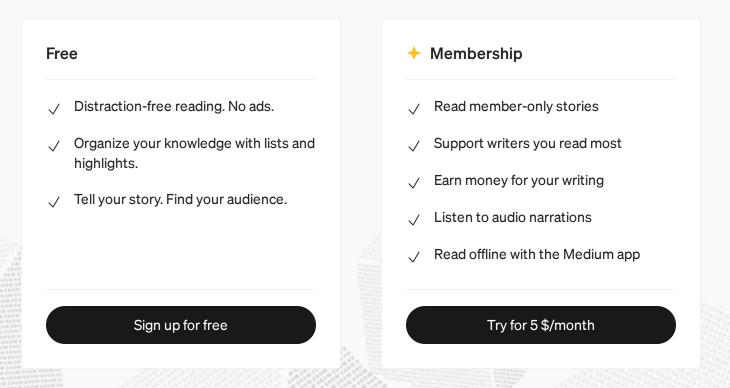
Introduction to ETW and event logging

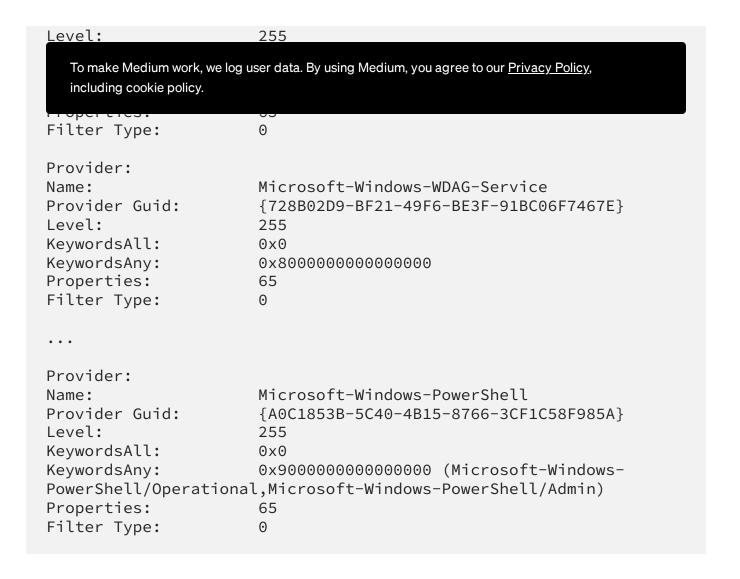
The <u>ETW architecture</u> differentiates between event *providers*, event *consumers*, and event *tracing sessions*. Tracing sessions are responsible for collecting events from providers and for relaying them to log files and consumers. Sessions are created and configured by *controllers* like the built-in <code>logman.exe</code> command line utility. Here are some useful commands for exploring existing trace sessions and their respective ETW providers; note that these must usually be executed from an elevated context.

List all running trace sessions

> logman query -ets		
Data Collector Set	Туре	Status
Circular Kernel Context Logger	Trace	Running
AppModel	Trace	Running
ScreenOnPowerStudyTraceSession	Trace	Running
DiagLog	Trace	Running
EventLog-Application	Trace	Running
EventLog-System	Trace	Running
LwtNetLog	Trace	Running
NtfsLog	Trace	Running
TileStore	Trace	Running
UBPM	Trace	Running
WdiContextLog	Trace	Running
WiFiSession	Trace	Running
UserNotPresentTraceSession	Trace	Running
Diagtrack-Listener	Trace	Running
MSDTC_TRACE_SESSION	Trace	Running
WindowsUpdate_trace_log	Trace	Running

Medium

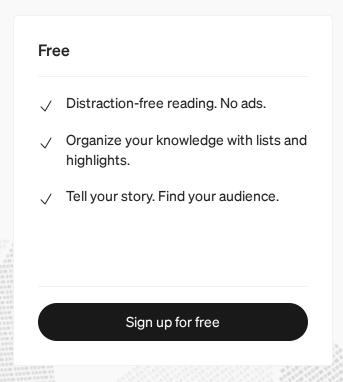


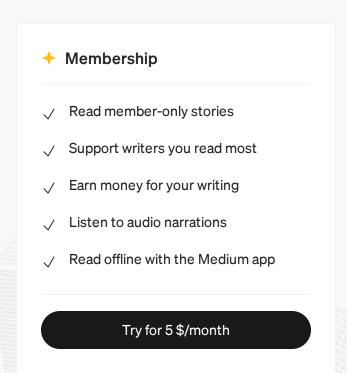


This command details the configuration of the trace session itself, followed by the configuration of each provider that the session is subscribed to, including the following parameters:

- Name: The name of the provider. A provider only has a name if it has a <u>registered manifest</u>, but it always has a unique GUID.
- **Provider GUID:** The unique GUID for the provider. The GUID and/or name of a provider is useful when performing research or operations on a specific provider.
- Level: The logging level specified. Standard logging levels are: 0 Log

Medium





in awant log-related trace sessions wou will see the high hate

To make Medium work, we log user data. By using Medium, you agree to our <u>Privacy Policy</u>, including cookie policy.

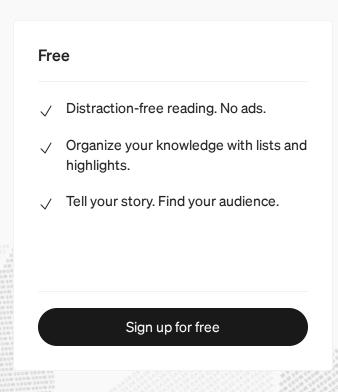
```
0x01 - Admin channel
0x02 - Debug channel
0x04 - Analytic channel
0x08 - Operational channel
```

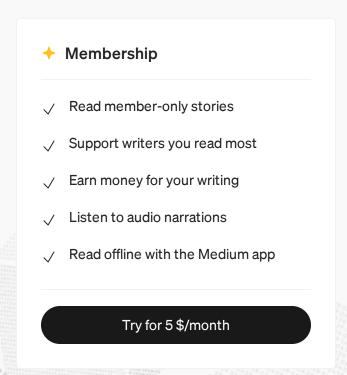
 Properties: This refers to optional ETW properties that can be specified when writing the event. The following values are currently supported (more information here):

```
0x001 - EVENT_ENABLE_PROPERTY_SID
0x002 - EVENT_ENABLE_PROPERTY_TS_ID
0x004 - EVENT_ENABLE_PROPERTY_STACK_TRACE
0x008 - EVENT_ENABLE_PROPERTY_PSM_KEY
0x010 - EVENT_ENABLE_PROPERTY_IGNORE_KEYWORD_0
0x020 - EVENT_ENABLE_PROPERTY_PROVIDER_GROUP
0x040 - EVENT_ENABLE_PROPERTY_ENABLE_KEYWORD_0
0x080 - EVENT_ENABLE_PROPERTY_PROCESS_START_KEY
0x100 - EVENT_ENABLE_PROPERTY_EVENT_KEY
0x200 - EVENT_ENABLE_PROPERTY_EXCLUDE_INPRIVATE
```

From a detection perspective, EVENT_ENABLE_PROPERTY_SID, EVENT_ENABLE_PROPERTY_TS_ID, EVENT_ENABLE_PROPERTY_PROCESS_START_KEY are valuable fields to collect. For example, EVENT_ENABLE_PROPERTY_PROCESS_START_KEY generates a value that uniquely identifies a process. Note that Process IDs are not unique identifiers for a process instance.

Medium





Enumerating all registered FTW providers

Th To make Medium work, we log user data. By using Medium, you agree to our <u>Privacy Policy</u>, including cookie policy.

Supplying their name and colorini bill provider to regional

binary manifest stored in the

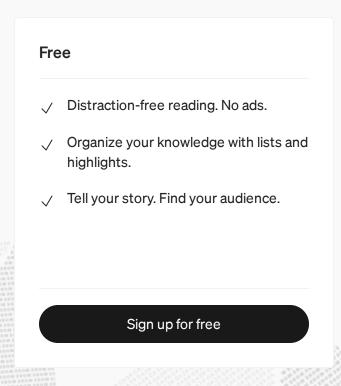
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\WINEVT\Publishers\
{PROVIDER_GUID} registry key. For example, the Microsoft-Windows-PowerShell provider has the following registry values:

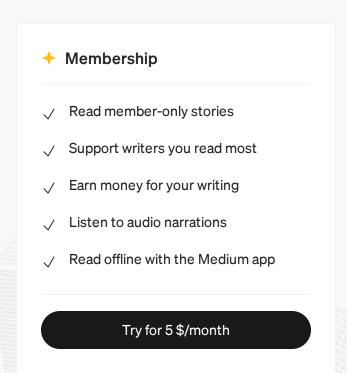
ETW and the event log know how to properly parse and display event information to a user based on binary-serialized information in the wevt_template resource present in the binaries listed in the ResourceFileName registry value. This resource is a binary representation of an instrumentation manifest (i.e., the schema for an ETW provider). The binary structure of wevt_template is under-documented, but there are at least two tools available to assist in parsing and recovering event schema, <u>WEPExplorer</u> and Perfview.

Viewing an individual provider

The logman tool prints basic information about a provider. For example:

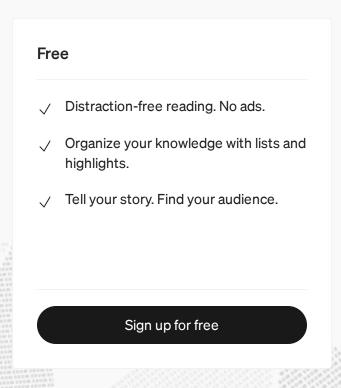
Medium

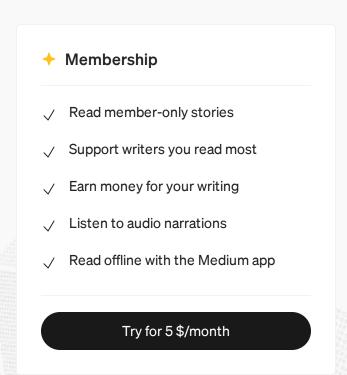




The listings shows supported keywords and logging values, as well as all processes that are registered to emit events via this provider. This output is useful for understanding how existing trace sessions filter on providers. It is also useful for initial discovery of potentially interesting information that could be gathered from via an ETW trace.

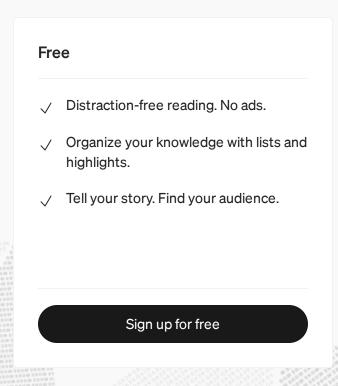
Medium

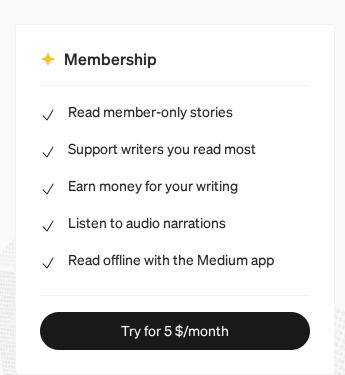




Entries listed with GUID are providers lacking a manifest. They will typically be related to <u>WPP</u> or <u>TraceLogging</u>, both of which are beyond the scope of this blog post. It is possible to retrieve provider names and event metadata for these providers types. For example, here are some of the resolved provider names from the unnamed providers above:

Medium





One of the great security features of PowerShell version 5 is scriptblock

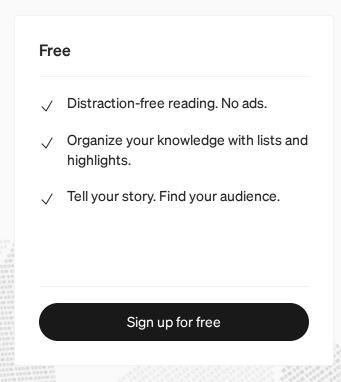
To make Medium work, we log user data. By using Medium, you agree to our Privacy Policy, including cookie policy.

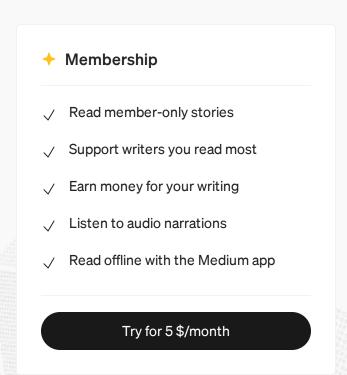
M:

(warning level) if the scriptblock contains any <u>suspicious terms</u>. The following C# <u>code</u> is executed to generate the event log:

From PowerShell

Medium

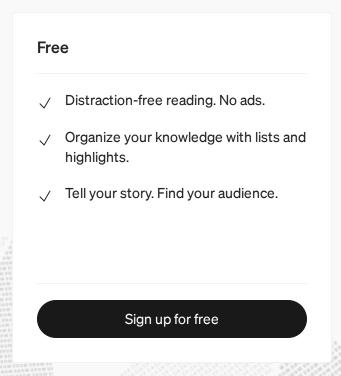


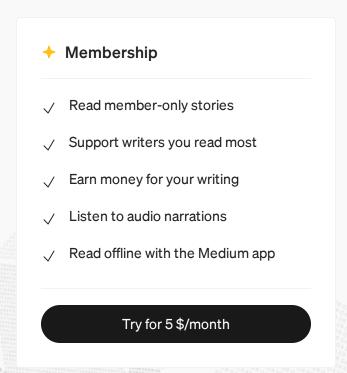


From PowerShell

The WriteEvent method is implemented as follows:

Medium



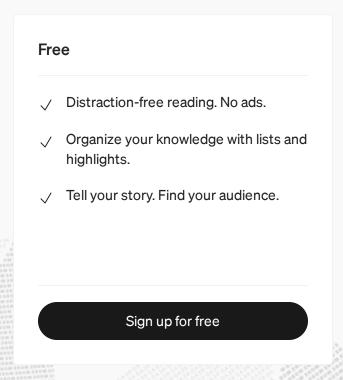


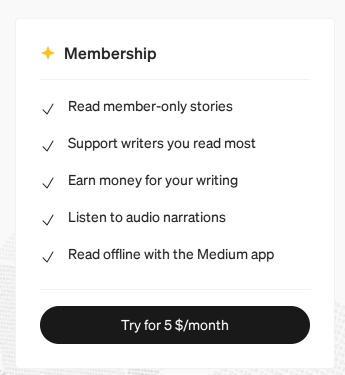
From PowerShell

Finally, the event information is marshaled and <u>EventWriteTransfer</u> is called, supplying the Microsoft-Windows-PowerShell provider with event data.

The relevant data supplied to EventWriteTransfer is as follows:

Medium





Normally, this event would not be logged but because the Application.

To make Medium work, we log user data. By using Medium, you agree to our Privacy Policy, including cookie policy.

which will log the event despite a keyword value not being specified.

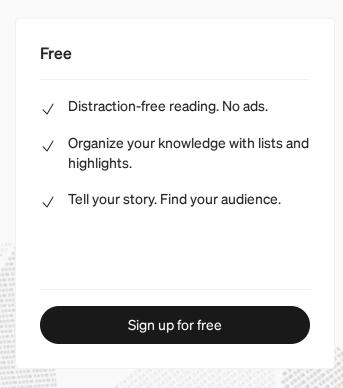
• Event data: the scriptblock contents and event fields

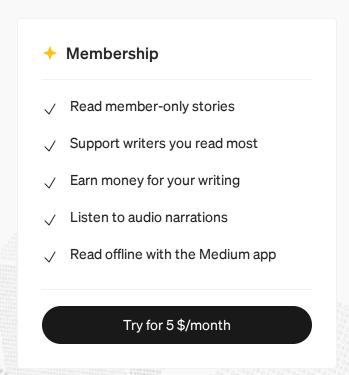
Upon receiving the event from the PowerShell ETW provider, the event log service parses the binary wevt_template schema (original XML schema) and presents human-readable, parsed event properties/fields:

amsi.dll event tracing

You may have observed that Windows 10 has an AMSI/Operational event log that is typically empty. To understand why events are not logged to this event log, you would first have to inspect how data is fed to the AMSI ETW provider (Microsoft-Antimalware-Scan-Interface - {2A576B87-09A7-520E-C21A-4942F0271D67}) and then observe how the Application event log trace session (EventLog-Application) subscribes to the AMSI ETW provider. Let's start by looking at the provider registration in the Application event log. The

Medium



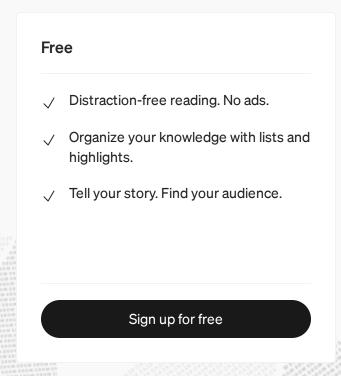


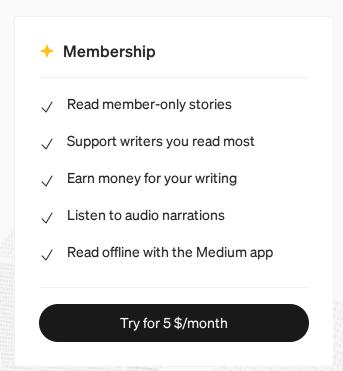
This information on its own does not explain why AMSI events are not logged, but it supplies needed context upon inspecting how amsi.dl writes events to ETW. By loading amsi.dl into IDA, we can see that there was a single call to the EventWrite function within the internal

CAmsiAntimalware::GenerateEtwEvent function:

The relevant portion of the call to EventWrite is the EventDescriptor

Medium



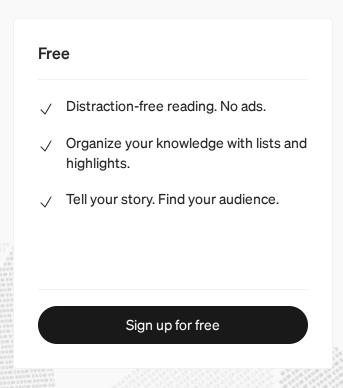


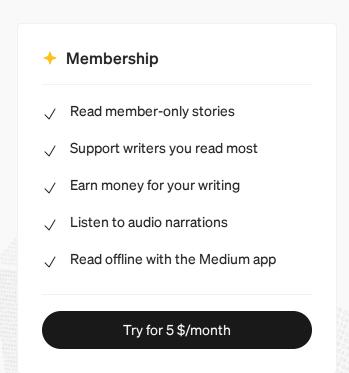
We now understand that 1101 events not logged to the Application event log because it only considers events where the keyword value matches 0x800000000000000. In order to fix this issue and get events pumping into the event log, either the Application event log trace session would need to be modified (not recommended and requires SYSTEM privileges) or you could create your own persistent trace session (e.g., an autologger) to capture AMSI events in the event log. The following PowerShell script creates such a trace session:

```
$AutoLoggerGuid = "{$((New-Guid).Guid)}"
New-AutologgerConfig -Name MyCustomAutoLogger -Guid
$AutoLoggerGuid -Start Enabled
Add-EtwTraceProvider -AutologgerName MyCustomAutoLogger -Guid
'{2A576B87-09A7-520E-C21A-4942F0271D67}' -Level 0xff -
MatchAnyKeyword 0x80000000000001 -Property 0x41
```

After running the above command, reboot, and the AMSI event log will begin to populate.

Medium





fed into the event log. One would have to recort to eneculation as to how the

To make Medium work, we log user data. By using Medium, you agree to our <u>Privacy Policy</u>, including cookie policy.

ETW tampering techniques

If the goal of an attacker is to subvert event logging, ETW provides a stealthy mechanism to affect logging without itself generating an event log trail. Below is a non-exhaustive list of tampering techniques that an attacker can use to cut off the supply of events to a specific event log.

Tampering techniques can generally be broken down into two categories:

- Persistent, requiring reboot i.e., a reboot must occur before the attack
 takes effect. Changes can be reverted, but would require another reboot.
 These attacks involve altering <u>autologger</u> settings persistent ETW trace
 sessions with settings in the registry. There are more types of persistent
 attacks than ephemeral attacks, and they are usually more
 straightforward to detect.
- 2. Ephemeral i.e., where the attack can take place without a reboot.

Autologger provider removal

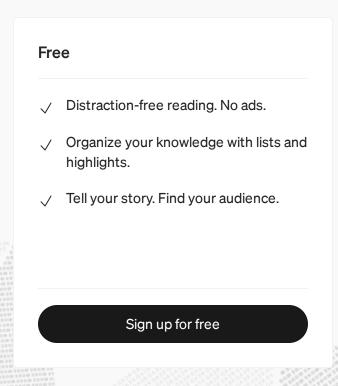
Tampering category: Persistent, requiring reboot Minimum permissions required: Administrator

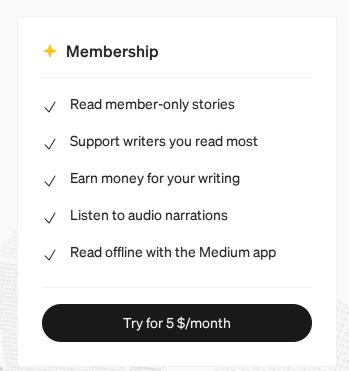
Detection artifacts: Registry key deletion:

HKLM\SYSTEM\CurrentControlSet\Control\WMI\Autologger\AUTOLOGGER_NAME\
{PROVIDER_GUID}

Description. This technique involves the removed of a provider entry from a

Medium





Tampering estagory: Dereistent requiring reheat

To make Medium work, we log user data. By using Medium, you agree to our <u>Privacy Policy</u>, including cookie policy.

HKLM\SYSTEM\CurrentControlSet\Control\WMI\Autologger\AUTOLOGGER_NAME\
{PROVIDER_GUID} - EnableProperty (REG_DWORD)

Description: This technique involves alerting the <code>Enable</code> keyword of an autologger session. For example, by default, all ETW provider entries in the <code>EventLog-Application</code> autologger session are set to <code>0x41</code> which translates to <code>EVENT_ENABLE_PROPERTY_SID</code> and <code>EVENT_ENABLE_PROPERTY_ENABLE_KEYWORD_0</code> is not documented; it specifies that any events generated for a provider should be logged even if the keyword value is set to 0. An attacker could swap out

EVENT_ENABLE_PROPERTY_ENABLE_KEYWORD_0 for

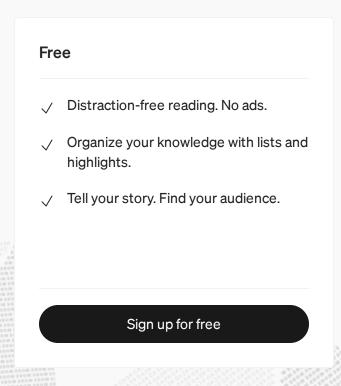
EVENT_ENABLE_PROPERTY_IGNORE_KEYWORD_0, resulting in a value of 0×11, which would result in all events where the keyword is 0 to not be logged. For example, PowerShell eventing supplies a 0 keyword value with its events, resulting in no logging to the PowerShell event log.

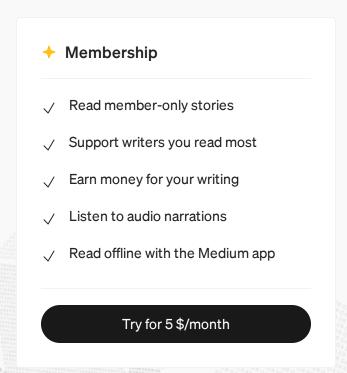
Example: The following PowerShell code disables Microsoft-Windows-PowerShell event logging:

```
Set-EtwTraceProvider -Guid '{A0C1853B-5C40-4B15-8766-3CF1C58F985A}' -AutologgerName 'EventLog-Application' -Property 0x11
```

In the above example, A0C1853B-5C40-4B15-8766-3CF1C58F985A refers to the Microsoft-Windows-PowerShell ETW provider. This command will end up

Medium





Tampering category: Enhanceal

De

To make Medium work, we log user data. By using Medium, you agree to our <u>Privacy Policy</u>, including cookie policy.

associated with this event. While the technique example below indicates that logman.exe was used to perform the attack, an attacker can obfuscate their techniques by using Win32 APIs directly, WMI, DCOM, PowerShell, etc.

Description: This technique involves removing an ETW provider from a trace session, cutting off its ability to supply a targeted event log with events until a reboot occurs, or until the attacker restores the provider. While an attacker must have SYSTEM privileges to perform this attack, it is unlikely that defenders will notice such an attack if they rely on event logs for threat detection.

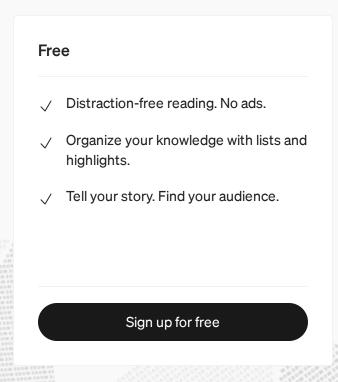
Example: The following PowerShell code immediately disables Microsoft-Windows-PowerShell event logging until a reboot occurs or the attacker restores the ETW provider:

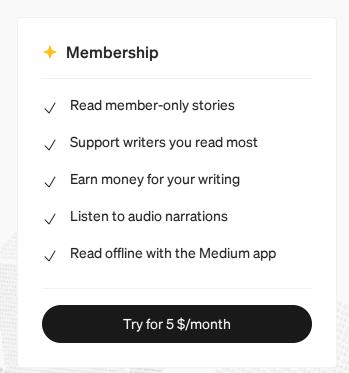
logman update trace EventLog-Application --p Microsoft-Windows-PowerShell -ets

Alternative detection artifacts/ideas:

• Event ID 12 within the Microsoft-Windows-Kernel-EventTracing/Analytic log indicates when a trace session is modified, but it doesn't supply the provider name or GUID that was removed, so it would be difficult to confidently determine whether or not something suspicious occurred

Medium





• The frequency at which providers are removed from Application event logs in large environments is not currently known. As as fallback, it is still advised to log the execution of <code>logman.exe</code>, <code>wpr.exe</code>, and PowerShell in your environment.

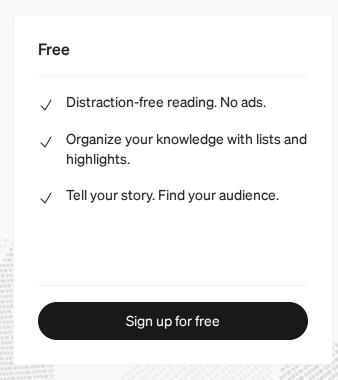
Conclusion

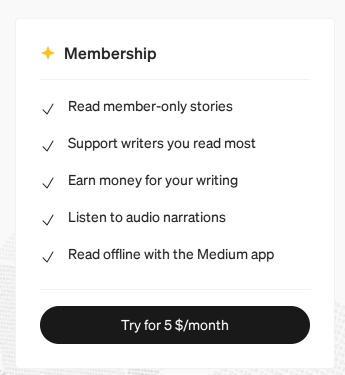
Identifying blind spots and assumptions in Alerting and Detection Strategies is a crucial step in ensuring the resilience of detections. Since ETW is at the core of the event logging infrastructure, gaining an in-depth understanding of ETW tampering attacks is a valuable way to increase the integrity of security-related data sources.

Further Reading

- <u>ETW Overview</u>
- <u>Instrumenting Your Code with ETW</u>
- Event Tracing for Windows: Reducing Everest to Pike's Peak

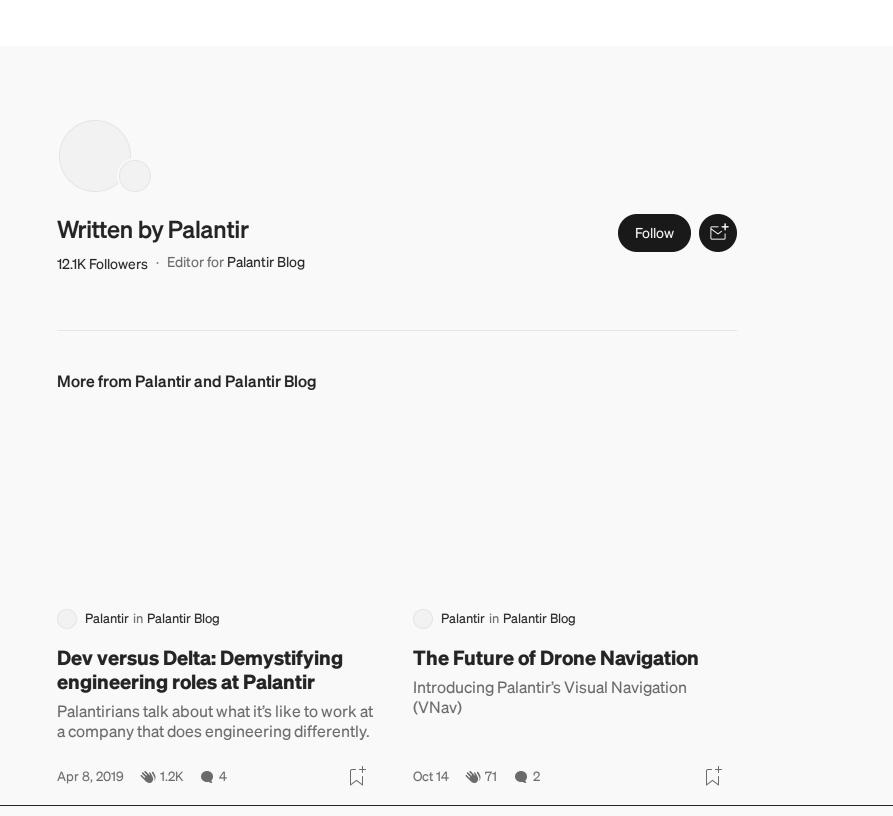
Medium



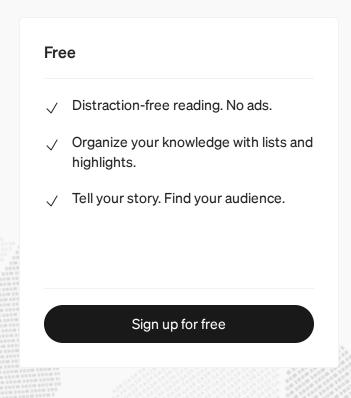


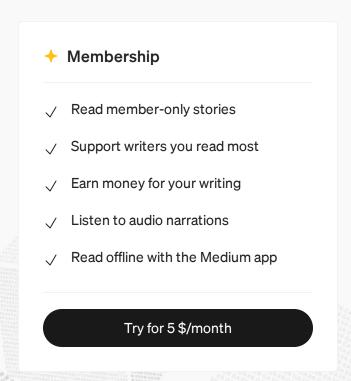
including cookie policy.

To make Medium work, we log user data. By using Medium, you agree to our Privacy Policy,

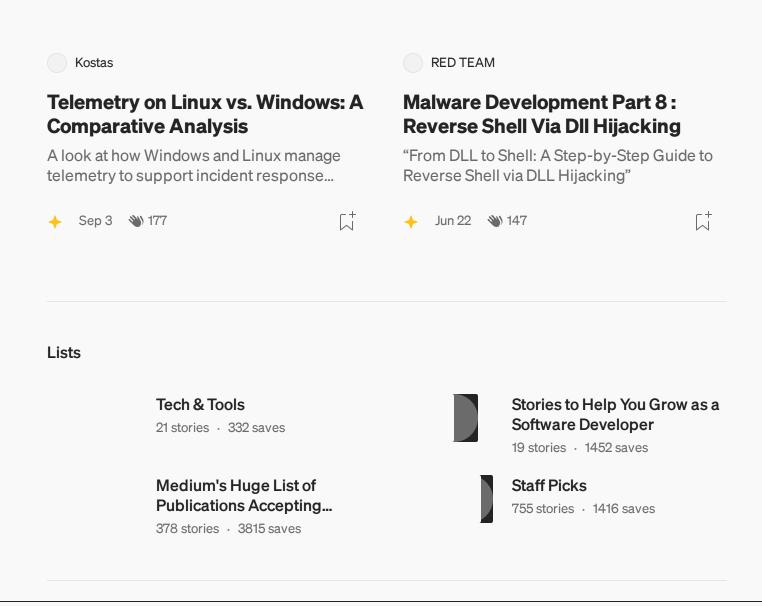


Medium

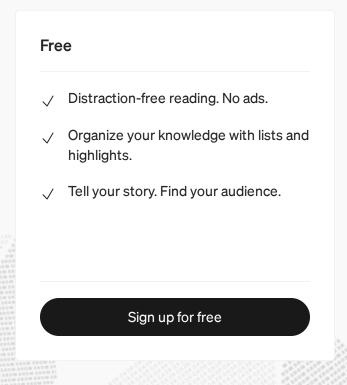


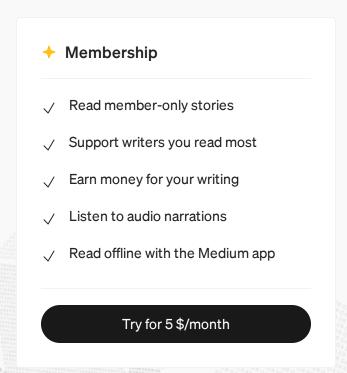


Recommended from Medium



Medium





To make Medium work, we log user data. By using Medium, you agree to our Privacy Policy, including cookie policy. Tamir Suliman Chicken0248 **Comparing Popular SIEM Data** [LetsDefend Write-up] Log Pipeline Designs: Elastic, ArcSig... **Analysis With Sysmon** We will continue with what we discussed on Our company has experienced a breach on Part 2. In this article we will provide breif... one of its endpoints. Your task is to investiga... \Box Sep 29 🔌 9 🗨 1 Sep 2 👋 3 See more recommendations Help Status About Careers Press Blog Privacy Terms Text to speech Teams

Medium

