

Event Tracing for Windows Internals

Shusei Tomonaga



Today's presentation slides here.

Motivation

Question

Can we use **Event Log** to detect all attacks?

Motivation

Question

Can we use **Event Log** to detect all attacks?

Answer

NO!

Motivation

Question

Can we use **Event Log** to detect all attacks?

Answer

NO!

We are looking for more **effective solutions.**

Motivation

Is there any log more detailed than Event Log on Windows OS?



Event Tracing for Windows

Goal of This Presentation

Understand the internals of the Event Tracing for Windows (ETW) to take your incident response to the next level.

Presentation Topics

1

What is ETW?

2

ETW Internals

3

ETW using Incident Response

4

Attack Surface

5

Mitigation and Detection

1

What is ETW?

2

ETW Internals

3

ETW using Incident Response

4

Attack Surface

5

Mitigation and Detection

What is Event Tracing for Windows

Event Tracing for Windows 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 or to determine where performance issues are occurring in the application.

What is Event Tracing for Windows

Uses of ETW

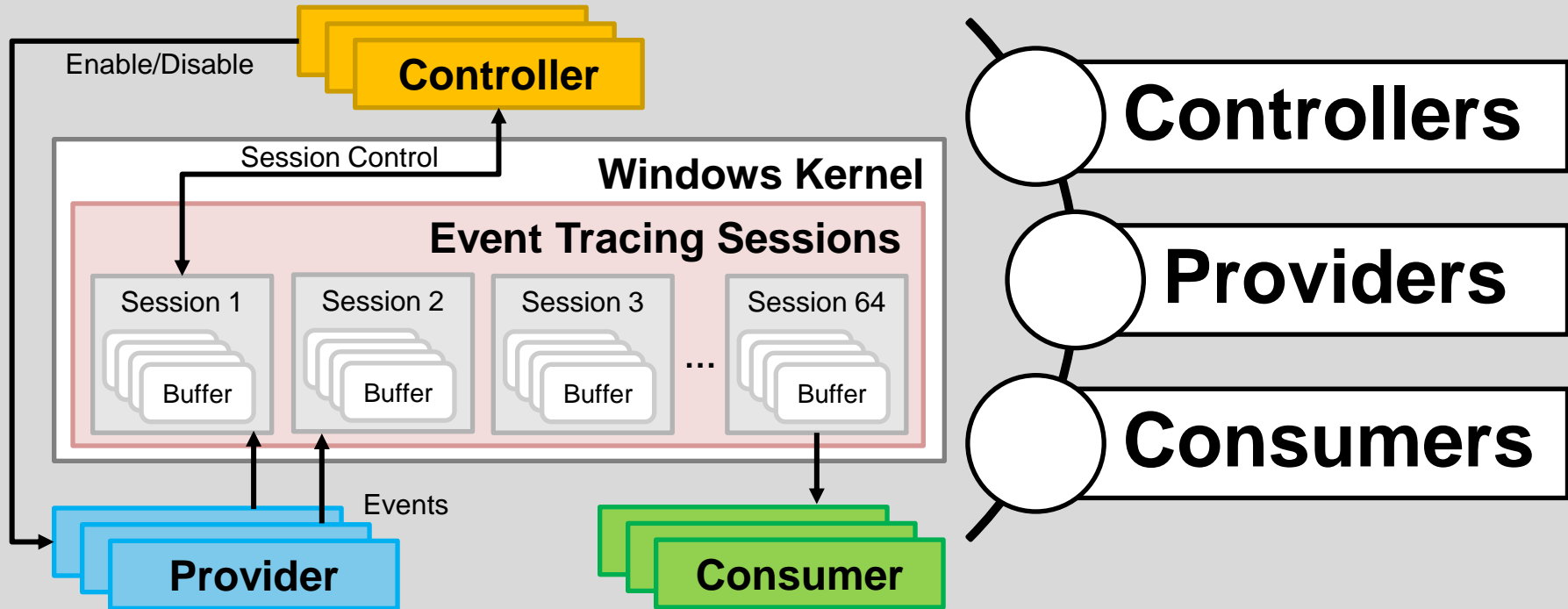
Application debugging

EDR

Event Log

What is Event Tracing for Windows

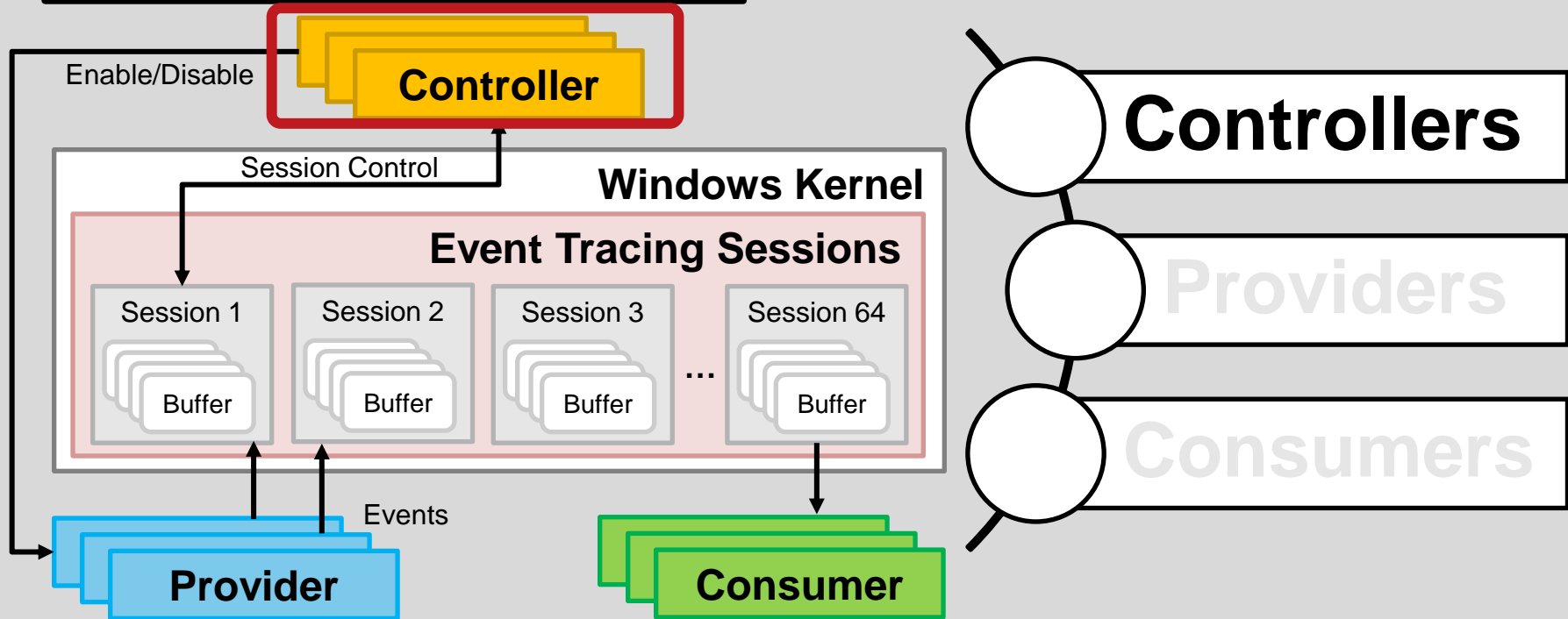
ETW has three functions



<https://learn.microsoft.com/ja-jp/windows/win32/etw/about-event-tracing>

What is Event Tracing for Windows

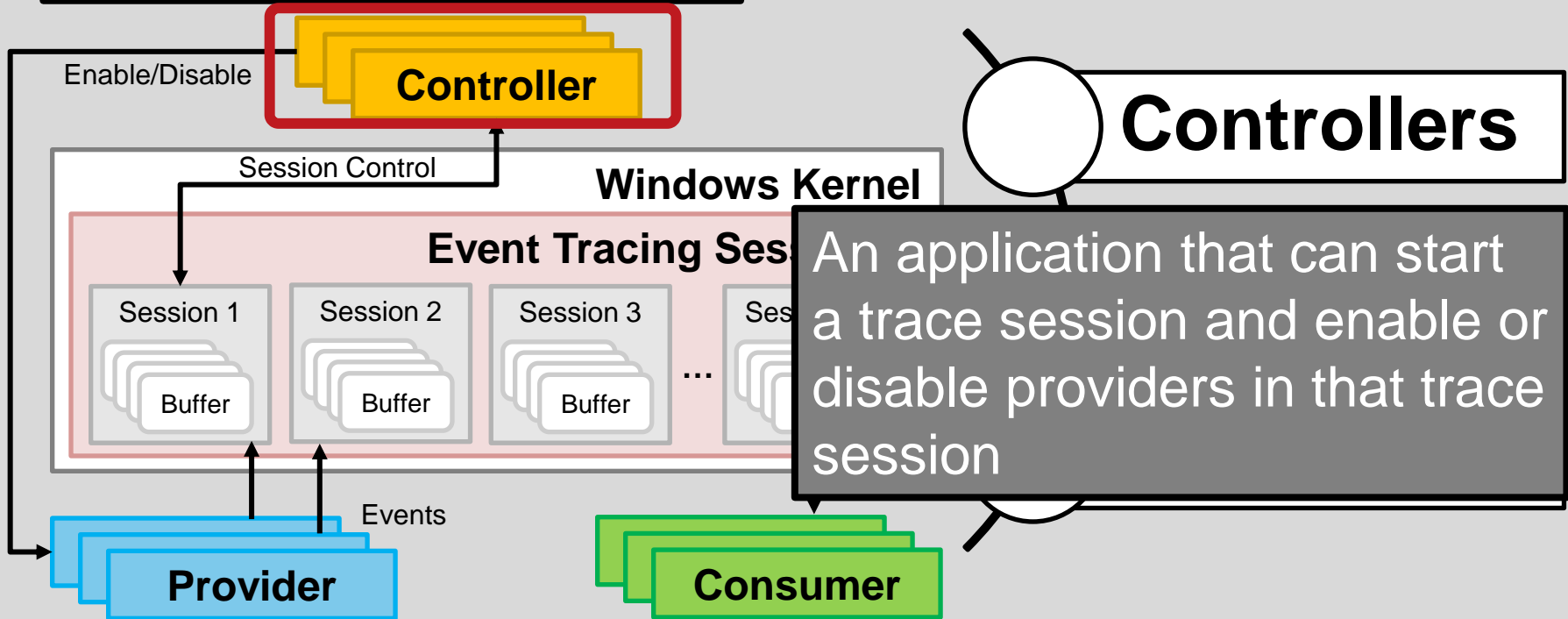
ETW has three functions



<https://learn.microsoft.com/ja-jp/windows/win32/etw/about-event-tracing>

What is Event Tracing for Windows

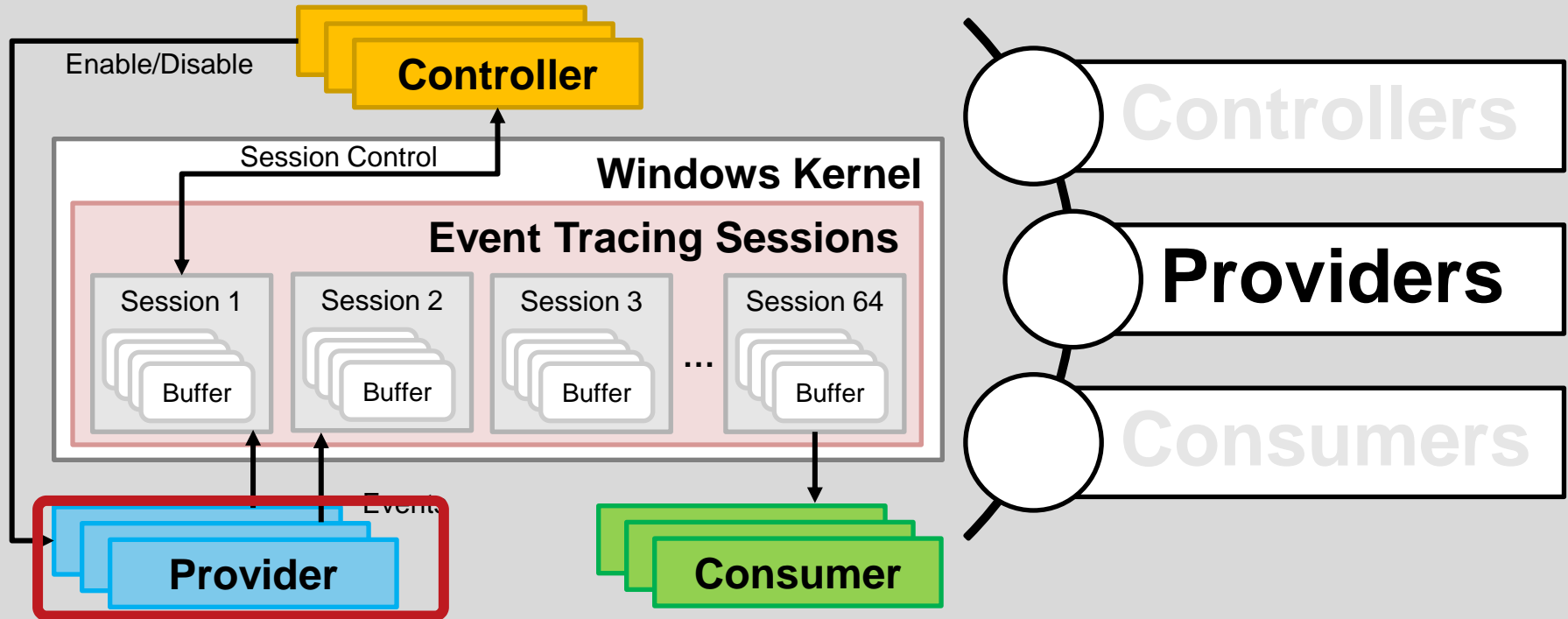
ETW has three functions



<https://learn.microsoft.com/ja-jp/windows/win32/etw/about-event-tracing>

What is Event Tracing for Windows

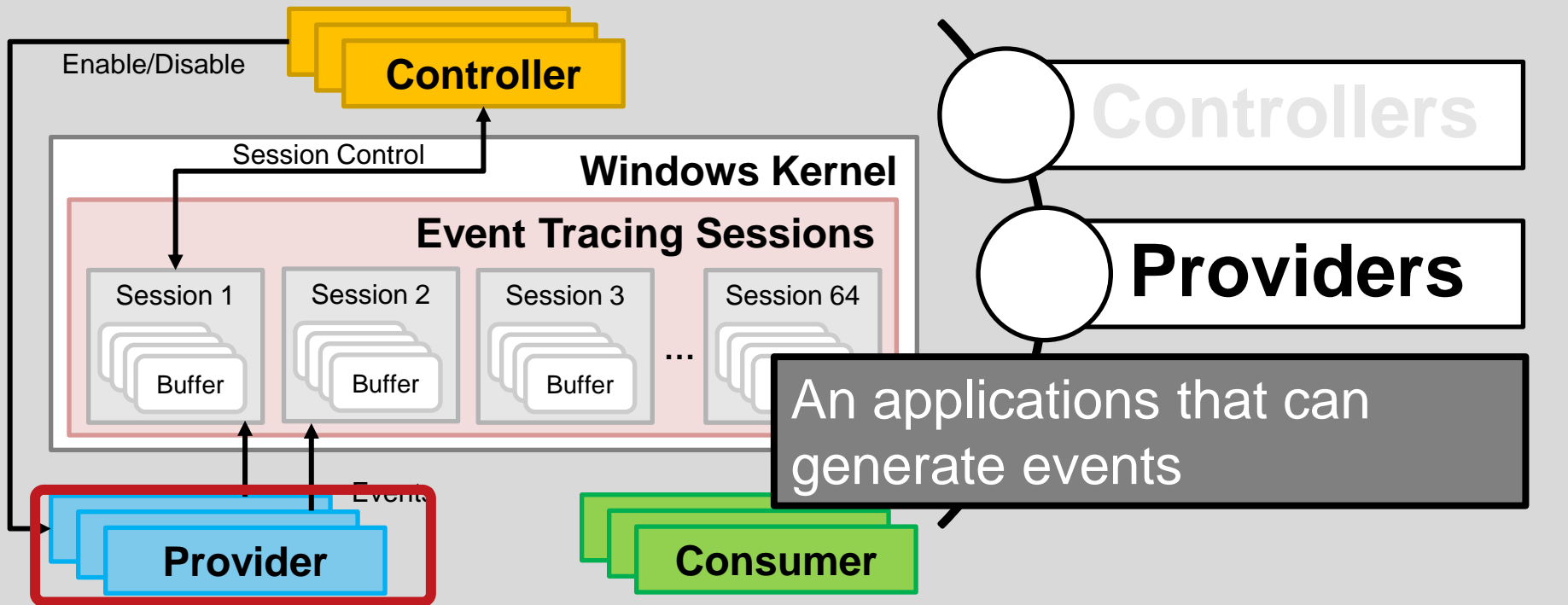
ETW has three functions



<https://learn.microsoft.com/ja-jp/windows/win32/etw/about-event-tracing>

What is Event Tracing for Windows

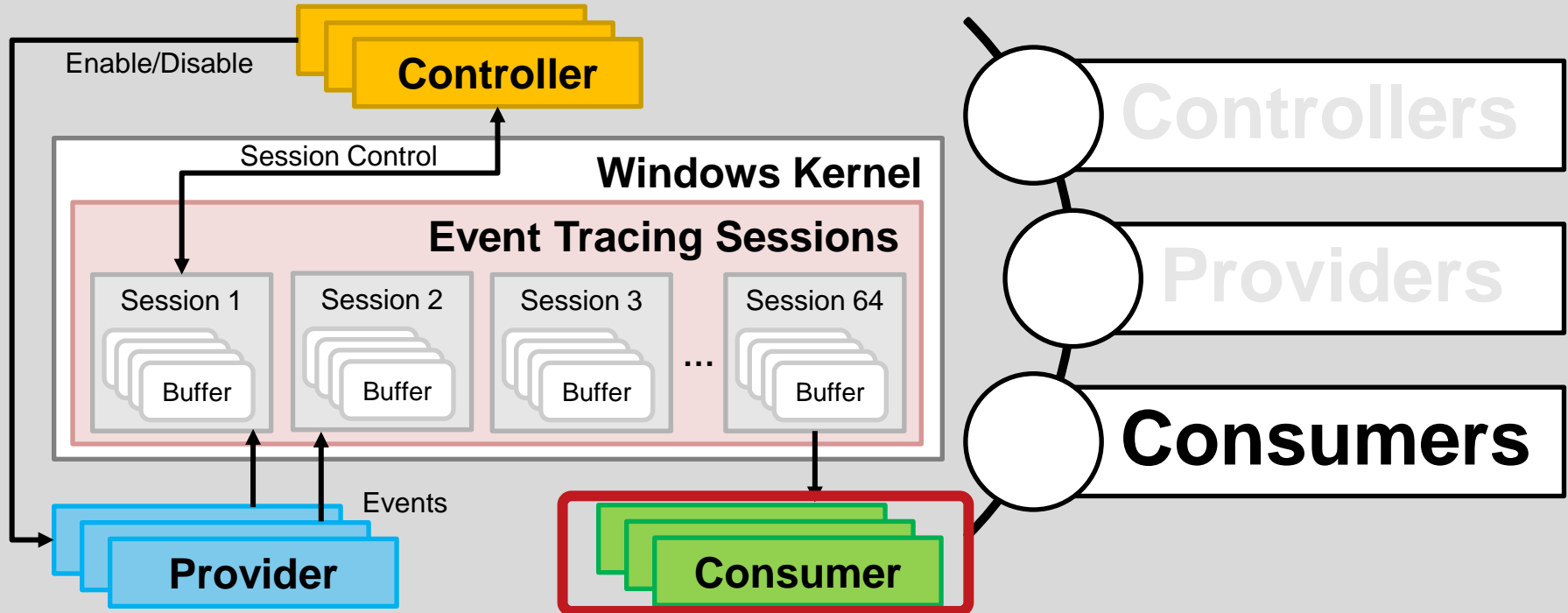
ETW has three functions



<https://learn.microsoft.com/ja-jp/windows/win32/etw/about-event-tracing>

What is Event Tracing for Windows

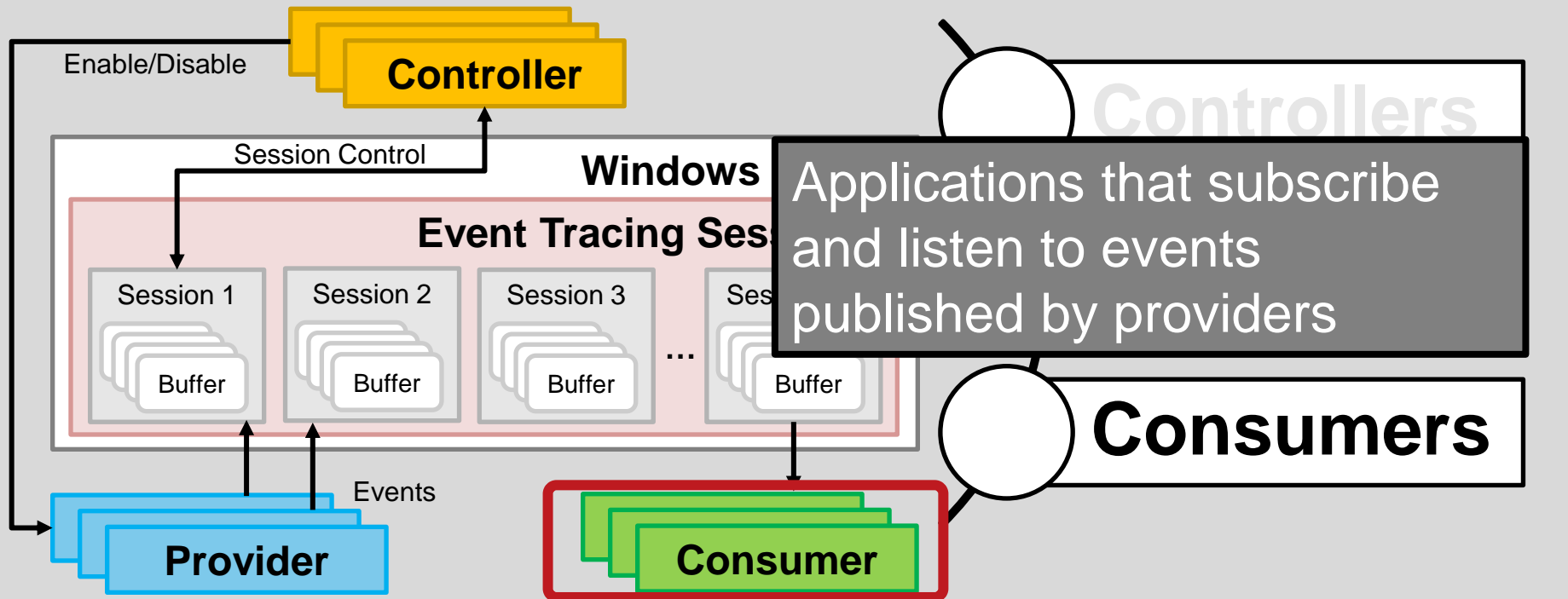
ETW has three functions



<https://learn.microsoft.com/ja-jp/windows/win32/etw/about-event-tracing>

What is Event Tracing for Windows

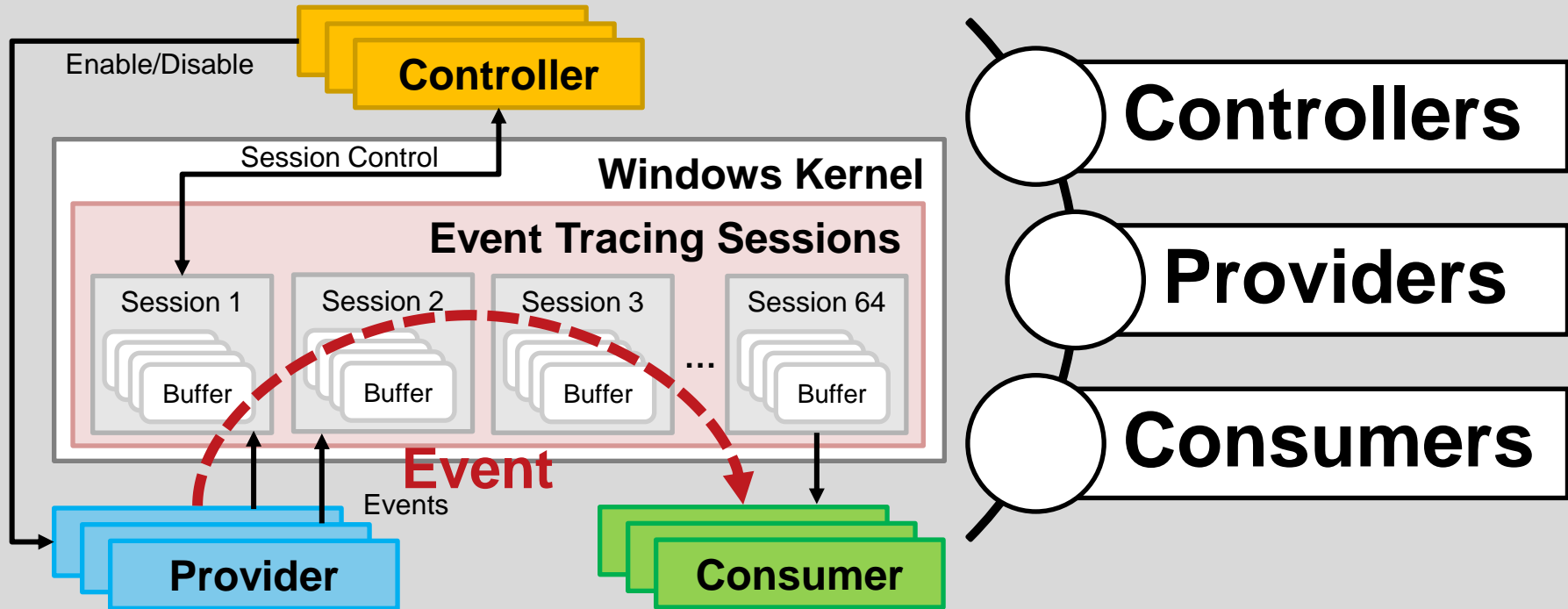
ETW has three functions



<https://learn.microsoft.com/ja-jp/windows/win32/etw/about-event-tracing>

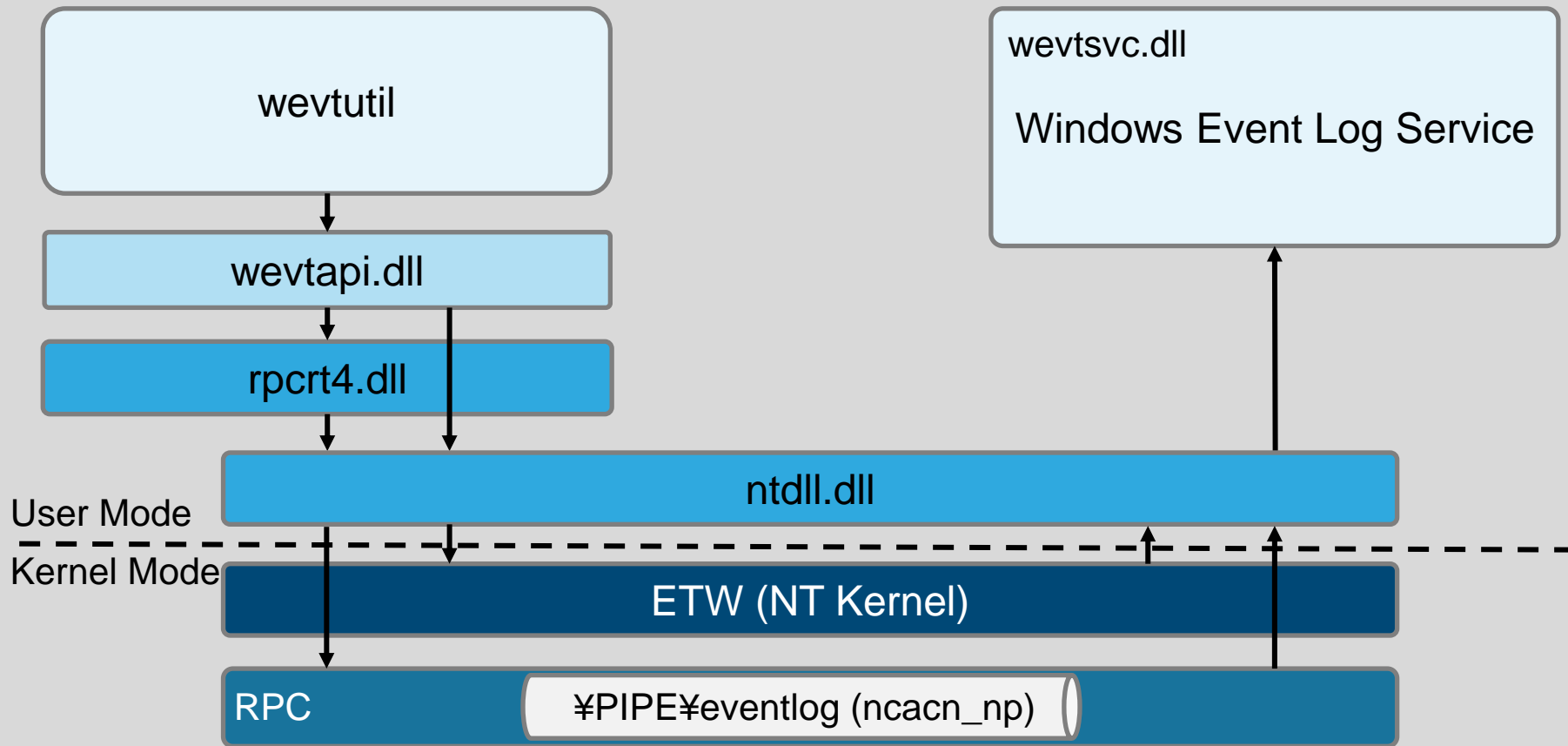
What is Event Tracing for Windows

ETW has three functions

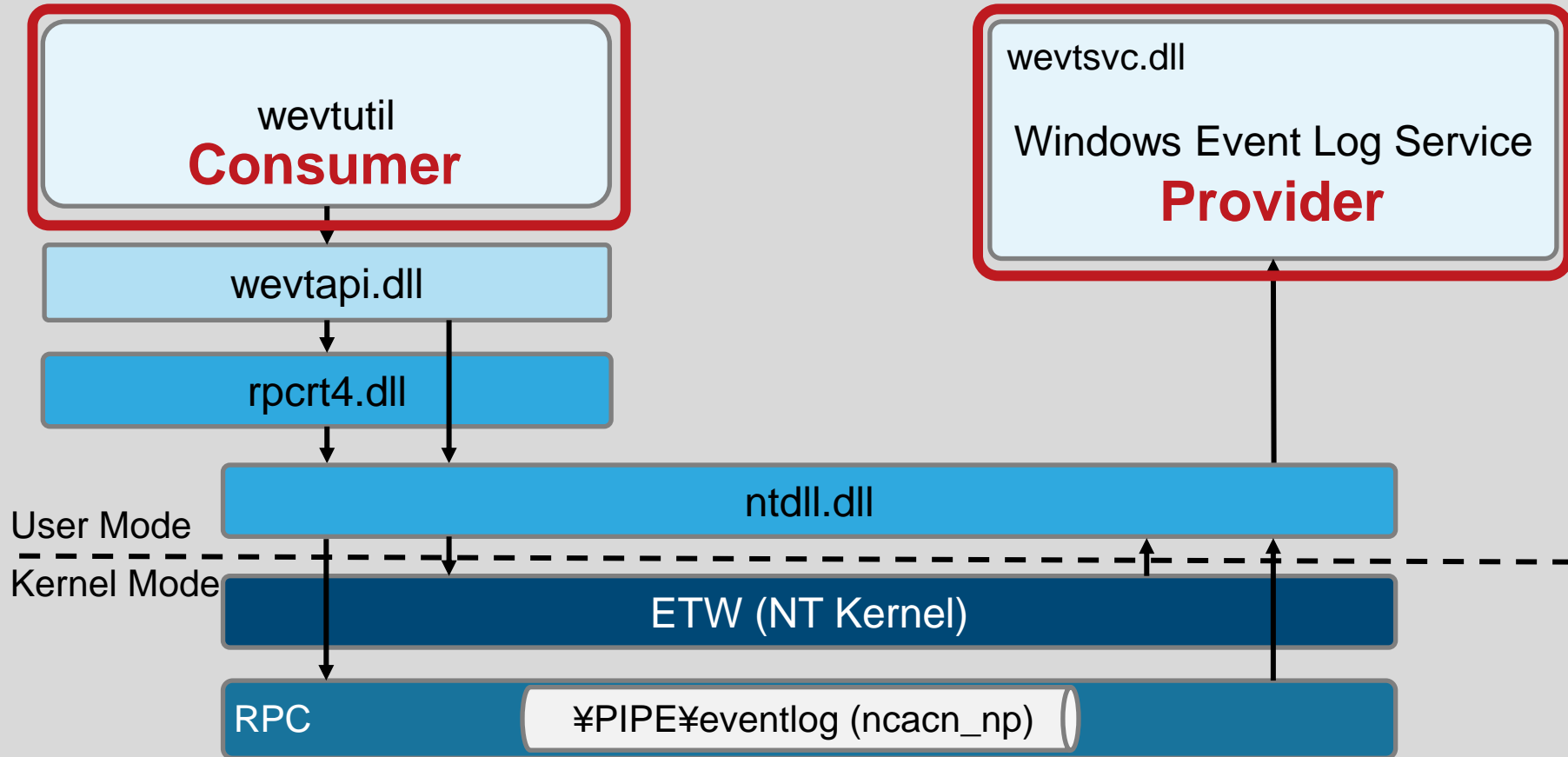


<https://learn.microsoft.com/ja-jp/windows/win32/etw/about-event-tracing>

Windows Event Log Control Flow using ETW

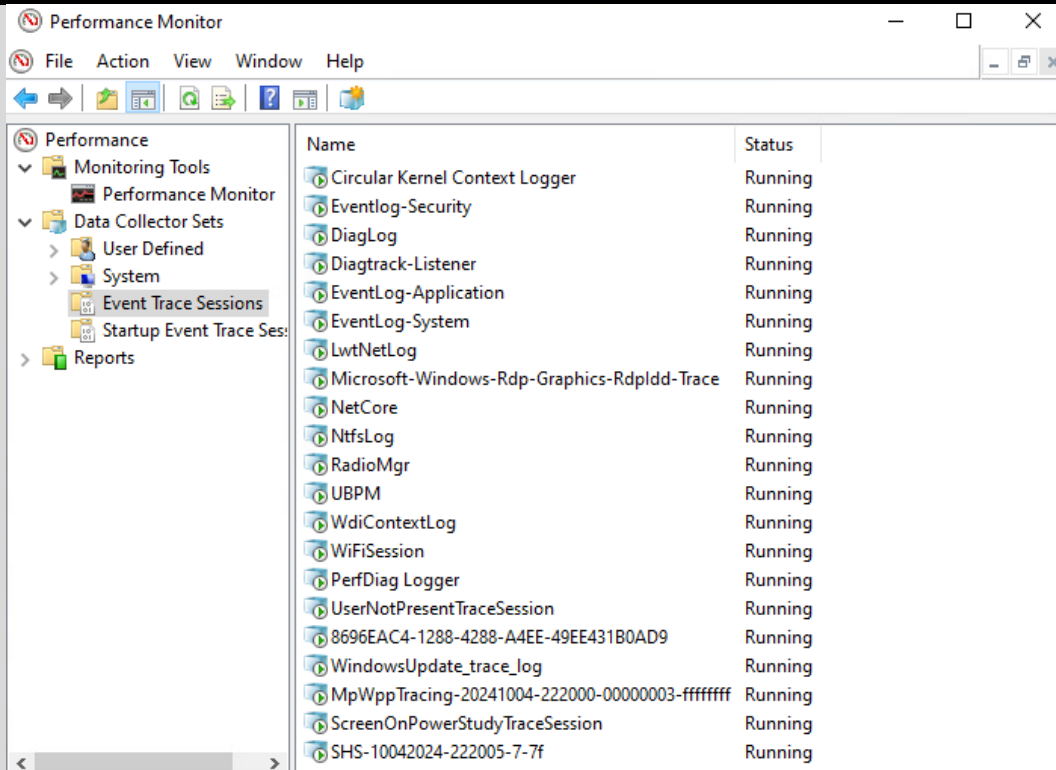


Windows Event Log Control Flow using ETW



How to Find ETW Sessions

Performance Monitor (perfmon) > Data Collector Set > Event Trace Sessions

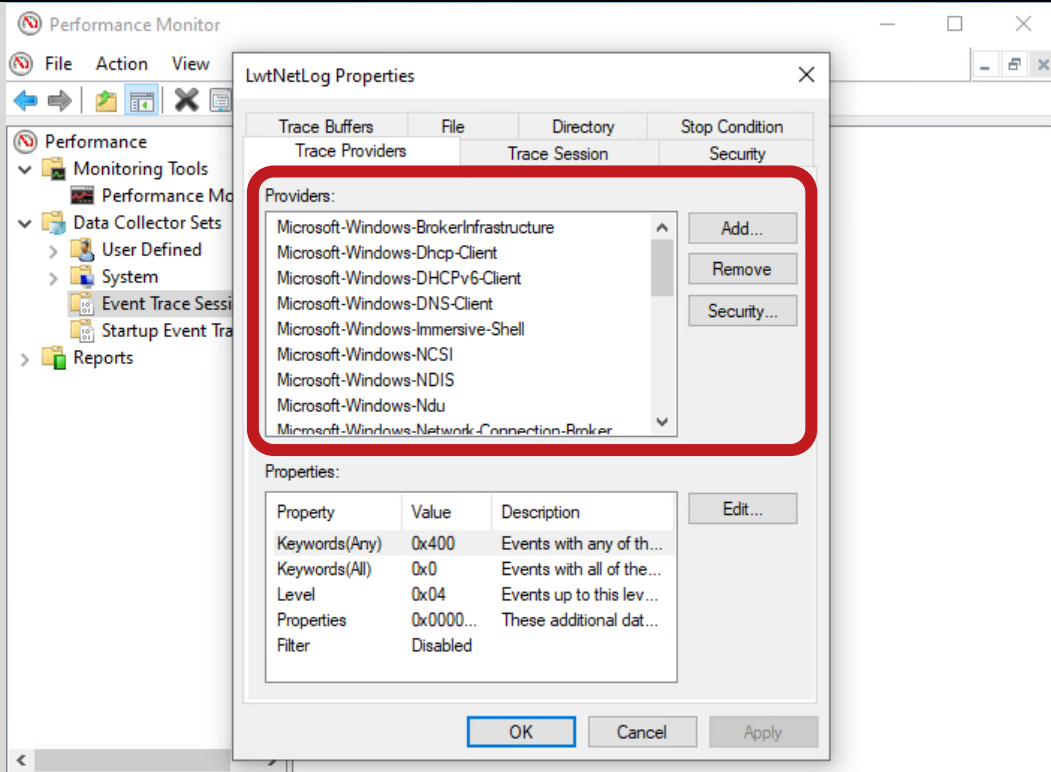


The screenshot shows the Performance Monitor application window. The left-hand tree view is expanded to 'Performance' > 'Data Collector Sets' > 'System' > 'Event Trace Sessions'. The main pane displays a list of 20 event trace sessions, all of which are in a 'Running' status.

Name	Status
Circular Kernel Context Logger	Running
Eventlog-Security	Running
DiagLog	Running
Diagtrack-Listener	Running
EventLog-Application	Running
EventLog-System	Running
LwtNetLog	Running
Microsoft-Windows-Rdp-Graphics-Rdpldd-Trace	Running
NetCore	Running
NtfsLog	Running
RadioMgr	Running
UBPM	Running
WdiContextLog	Running
WiFiSession	Running
PerfDiag Logger	Running
UserNotPresentTraceSession	Running
8696EAC4-1288-4288-A4EE-49EE431B0AD9	Running
WindowsUpdate_trace_log	Running
MpWppTracing-20241004-222000-00000003-ffffff	Running
ScreenOnPowerStudyTraceSession	Running
SHS-10042024-222005-7-7f	Running

How to Find ETW Sessions

Performance Monitor (perfmon) > Data Collector Set > Event Trace Sessions



How to Find ETW Sessions Using Command-line

> logman query -ets

Administrator: Command Prompt

```
C:\Windows\system32>logman query -ets
```

Data Collector Set	Type	Status
Circular Kernel Context Logger	Trace	Running
Eventlog-Security	Trace	Running
DiagLog	Trace	Running
Diagtrack-Listener	Trace	Running
EventLog-Application	Trace	Running
EventLog-System	Trace	Running
LwtNetLog	Trace	Running
Microsoft-Windows-Rdp-Graphics-RdpIdd-Trace	Trace	Running
NetCore	Trace	Running
NtfsLog	Trace	Running
RadioMgr	Trace	Running
UBPM	Trace	Running
WdiContextLog	Trace	Running

How to Find ETW Providers

> logman query providers

Administrator: Command Prompt

```
C:\Windows\system32>logman query providers
```

Provider	GUID
ACPI Driver Trace Provider	{DAB01D4D-2D48-477D-B1C3-DAAD0CE6F06B}
Active Directory Domain Services: SAM	{8E598056-8993-11D2-819E-0000F875A064}
Active Directory: Kerberos Client	{BBA3ADD2-C229-4CDB-AE2B-57EB6966B0C4}
Active Directory: NetLogon	{F33959B4-DBEC-11D2-895B-00C04F79AB69}
ADODB.1	{04C8A86F-3369-12F8-4769-24E484A9E725}
ADOMD.1	{7EA56435-3F2F-3F63-A829-F0B35B5CAD41}
Application Popup	{47BFA2B7-BD54-4FAC-B70B-29021084CA8F}
Application-Addon-Event-Provider	{A83FA99F-C356-4DED-9FD6-5A5EB8546D68}
ATA Port Driver Tracing Provider	{D08BD885-501E-489A-BAC6-B7D24BFE6BBF}
AuthFw NetShell Plugin	{935F4AE6-845D-41C6-97FA-380DAD429B72}
BCP.1	{24722B88-DF97-4FF6-E395-DB533AC42A1E}
BFE Trace Provider	{106B464A-8043-46B1-8CB8-E92A0CD7A560}
BITS Service Trace	{4A8AAA94-CFC4-46A7-8E4E-17BC45608F0A}
Certificate Services Client CredentialRoaming Trace	{EF4109DC-68FC-45AF-B329-CA2825437209}

How to Find ETW Providers

> logman query providers

Administrator: Command Prompt

```
C:\Windows\system32>logman query providers
```

Provider	GUID
ACPI Driver Trace Provider	{DAB01D4D-2D48-477D-B1C3-DAAD0CE6F06B}
Active Directory Domain Services	
Active Directory: Kerberos	
Active Directory: NetLogon	
ADODB.1	
ADOMD.1	
Application Popup	
Application-Addon-Event Provider	{A037A891-6596-42E8-97D8-9A2E8978900}
ATA Port Driver Tracing Provider	{D08BD885-501E-489A-BAC6-B7D24BFE6BBF}
AuthFw NetShell Plugin	{935F4AE6-845D-41C6-97FA-380DAD429B72}
BCP.1	{24722B88-DF97-4FF6-E395-DB533AC42A1E}
BFE Trace Provider	{106B464A-8043-46B1-8CB8-E92A0CD7A560}
BITS Service Trace	{4A8AAA94-CFC4-46A7-8E4E-17BC45608F0A}
Certificate Services Client CredentialRoaming Trace	{EF4109DC-68FC-45AF-B329-CA2825437209}

Over 1,000

ETW Providers for Threat Intelligence

For Example

Microsoft-
Windows-
Threat-
Intelligence

Microsoft-
Windows-
Security-
Auditing

Microsoft-
Windows-
Kernel-Audit-
API-Calls

Microsoft-
Antimalware-
Protection

Microsoft-
Antimalware-
AMFilter

Microsoft-
Windows-
Eventlog

Microsoft-
Windows-
PowerShell

Microsoft-
Windows-WMI

ETW Providers for Threat Intelligence

For Example

Microsoft-
Windows-
Threat-
Intelligence

Microsoft-
Windows-
Security-
Auditing

**Microsoft-
Windows-
Kernel-Audit-
API-Calls**

Microsoft-
Antimalware-
Protection

Microsoft-
Antimalware-
AMFilter

Microsoft-
Windows-
Eventlog

Microsoft-
Windows-
PowerShell

Microsoft-
Windows-WMI

ETW Providers for Threat Intelligence

For Example

Microsoft-
Windows-
Threat-
Intelligence

Microsoft-
Windows-
Security-
Auditing

Microsoft-
Windows-
Kernel-Audit-
API-Calls

**Microsoft-
Antimalware-
Protection**

Microsoft-
Antimalware-
AMFilter

Microsoft-
Windows-
Eventlog

Microsoft-
Windows-
PowerShell

Microsoft-
Windows-WMI

ETW Providers for Threat Intelligence

For Example

Microsoft-
Windows-
Threat-
Intelligence

Microsoft-
Windows-
Security-
Auditing

Microsoft-
Windows-
Kernel-Audit-
API-Calls

Microsoft-
Antimalware-
Protection

Microsoft-
Antimalware-
AMFilter

Microsoft-
Windows-
Eventlog

**Microsoft-
Windows-
PowerShell**

Microsoft-
Windows-WMI

Microsoft-Windows-Threat-Intelligence

```
C:\>logman query providers "Microsoft-Windows-Threat-Intelligence"
```

Provider	GUID
Microsoft-Windows-Threat-Intelligence	{F4E1897C-BB5D-5668-F1D8-040F4D8DD344}

Value	Keyword	Description
0x0000000000000001	KERNEL_THREATINT_KEYWORD_ALLOCVM_LOCAL	
0x0000000000000002	KERNEL_THREATINT_KEYWORD_ALLOCVM_LOCAL_KERNEL_CALLER	
0x0000000000000004	KERNEL_THREATINT_KEYWORD_ALLOCVM_REMOTE	
0x0000000000000008	KERNEL_THREATINT_KEYWORD_ALLOCVM_REMOTE_KERNEL_CALLER	
...		
0x0000000000001000	KERNEL_THREATINT_KEYWORD_QUEUEUSERAPC_REMOTE	
0x0000000000002000	KERNEL_THREATINT_KEYWORD_QUEUEUSERAPC_REMOTE_KERNEL_CALLER	
0x0000000000004000	KERNEL_THREATINT_KEYWORD_SETTHREADCONTEXT_REMOTE	
...		
0x000000000100000	KERNEL_THREATINT_KEYWORD_SUSPEND_THREAD	
0x000000000200000	KERNEL_THREATINT_KEYWORD_RESUME_THREAD	
0x000000000400000	KERNEL_THREATINT_KEYWORD_SUSPEND_PROCESS	
0x000000000800000	KERNEL_THREATINT_KEYWORD_RESUME_PROCESS	

ETW Providers used by AV/EDR

Research AV/EDR

- Windows Defender
- Kaspersky
- ESET
- TrendMicro
- Symantec
- McAfee
- Cylance
- Filebeat (Elastic)

ETW Providers used by AV/EDR

Research AV/EDR

- Windows Defender
- Kaspersky
- ESET
- TrendMicro
- Symantec
- McAfee
- Cylance
- Filebeat (Elastic)



```
Microsoft-Windows-AppModel-Runtime
Microsoft-Windows-AppxPackagingOM
Microsoft-Windows-CAPI2
Microsoft-Windows-Crypto-NCrypt
Microsoft-Windows-Deplorch
Microsoft-Windows-DNS-Client
Microsoft-Windows-Eventlog
Microsoft-Windows-KnownFolders
Microsoft-Windows-LDAP-Client
Microsoft-Windows-NetworkProfile
Microsoft-Windows-Perflib
Microsoft-Windows-PrintService
Microsoft-Windows-RestartManager
Microsoft-Windows-RPC-Events
Microsoft-Windows-Shell-Core
Microsoft-Windows-User Profiles General
Microsoft-Windows-UserPnp
Microsoft-Windows-VerifyHardwareSecurity
Microsoft-Windows-WinHttp
Microsoft-Windows-WinINet-Pca
Network Location Awareness Trace
Network Profile Manager
WLAN Diagnostics Trace
```

ETW Providers used by AV/EDR

Research AV/EDR

- Windows Defender
- Kaspersky
- ESET
- TrendMicro
- Symantec
- McAfee
- Cylance
- Filebeat (Elastic)



```
Microsoft-Windows-AppModel-Runtime
Microsoft-Windows-CAPI2
Microsoft-Windows-Crypto-NCrypt
Microsoft-Windows-Deplorch
Microsoft-Windows-DNS-Client
Microsoft-Windows-Eventlog
Microsoft-Windows-Kernel-AppCompat
Microsoft-Windows-KnownFolders
Microsoft-Windows-LDAP-Client
Microsoft-Windows-Networking-Correlation
Microsoft-Windows-RPC
Microsoft-Windows-RPC-Events
Microsoft-Windows-Shell-Core
Microsoft-Windows-User Profiles General
Microsoft-Windows-UserPnp
Microsoft-Windows-WinHttp
Microsoft-Windows-WinINet-Pca
```

ETW Providers used by AV/EDR

Important ETW Provider List

Microsoft-Windows-AppModel-Runtime

Microsoft-Windows-CAPI2

Microsoft-Windows-COMRuntime

Microsoft-Windows-Crypto-NCrypt

Microsoft-Windows-Deplorch

Microsoft-Windows-DNS-Client

Microsoft-Windows-Eventlog

Microsoft-Windows-Kernel-AppCompat

Microsoft-Windows-KnownFolders

Microsoft-Windows-LDAP-Client

Microsoft-Windows-Networking-Correlation

Microsoft-Windows-NetworkProfile

Microsoft-Windows-PrintService

Microsoft-Windows-RPC

Microsoft-Windows-RPC-Events

Microsoft-Windows-Shell-Core

Microsoft-Windows-User Profiles General

Microsoft-Windows-UserPnp

Microsoft-Windows-VerifyHardwareSecurity

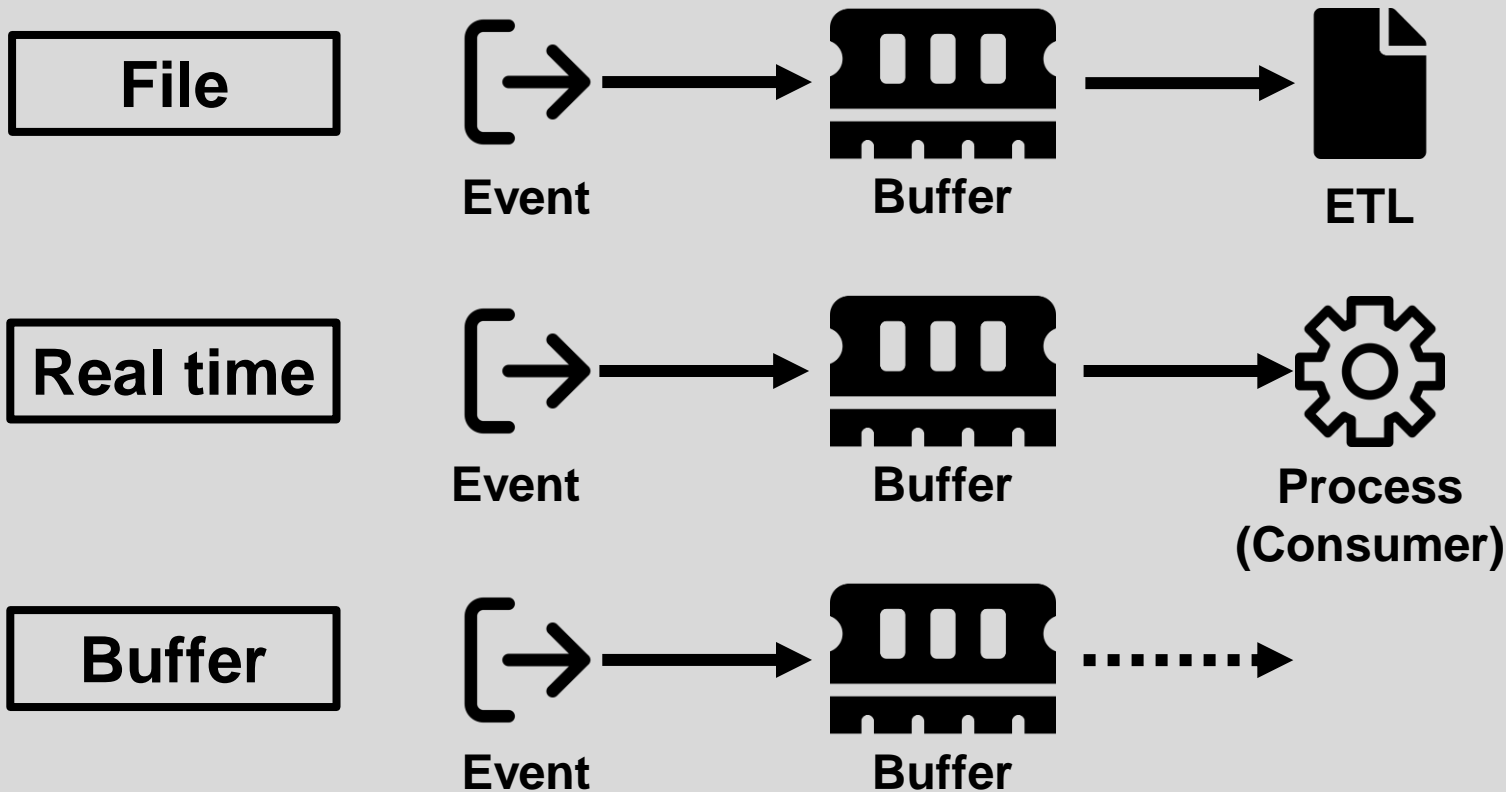
Microsoft-Windows-WinHttp

Microsoft-Windows-WinINet-Pca

Network Profile Manager

WLAN Diagnostics Trace

ETW Stream Mode



1

What is ETW?

2

ETW Internals

3

ETW using Incident Response

4

Attack Surface

5

Mitigation and Detection

What is ETL?

ETW events are saved in a ETL (Event Trace Log) file.

```
00000000 00 20 00 00 60 02 00 00 60 02 00 00 00 00 00 00 . . . . .
00000010 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 . . . . .
00000020 00 00 00 00 00 00 00 00 00 00 26 00 03 00 00 00 . . . . .
00000030 60 02 00 00 21 00 04 00 00 00 00 00 00 00 00 00 `...!.....
00000040 00 00 00 00 00 00 00 00 02 00 02 C0 C2 01 00 00 . . . . .AA..
00000050 44 3B 00 00 18 29 00 00 95 CF 9F 3D 71 0B 00 00 D;...)..•İY=q...
00000060 5C 08 00 00 4D 04 00 00 20 00 00 0A 00 01 05 \...M.....
00000070 65 4A 00 00 08 00 00 00 76 4E A7 59 78 8A DA 01 eJ.....vNSYxSÛ.
00000080 5A 62 02 00 00 00 00 00 05 00 00 00 01 00 00 00 Zb.....
00000090 01 00 00 00 08 00 00 00 00 00 00 00 00 09 00 00 . . . . .
000000A0 09 00 00 00 00 00 00 00 06 00 00 00 00 00 00 00 . . . . .
000000B0 E4 FD FF FF 40 00 74 00 7A 00 72 00 65 00 73 00 äÿÿÿ@.t.z.r.e.s.
000000C0 2E 00 64 00 6C 00 6C 00 2C 00 2D 00 36 00 33 00 ..d.l.l.,.-6.3.
000000D0 32 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 2.....
000000E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 . . . . .
000000F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 . . . . .
00000100 00 00 00 00 00 00 00 00 40 00 74 00 7A 00 72 00 . . . . .@.t.z.r.
00000110 65 00 73 00 2E 00 64 00 6C 00 6C 00 2C 00 2D 00 e.s...d.l.l.,.-
00000120 36 00 33 00 31 00 00 00 00 00 00 00 00 00 00 00 6.3.l.....
00000130 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 . . . . .
00000140 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 . . . . .
00000150 00 00 00 00 00 00 00 00 C4 FF FF FF 00 00 00 00 . . . . .ÿÿÿÿ
```

ETL files are in binary format and cannot be viewed without converting their contents.

ETL File Format

00000000	00 20 00 00 60 02 00 00 60 02 00 00 00 00 00 00
00000001	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000002	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000003	60 02 00 00 21 00 04 00 00 00 00 00 00 00 00 00!
00000004	00 00 00 00 00 00 00 00 02 00 02 C0 C2 01 00 00 00AA...
00000005	44 3B 00 00 18 29 00 00 95 CF 9F 3D 71 0B 00 00 00	D;...)*iY=q...
00000006	5C 08 00 00 4D 04 00 00 00 20 00 00 0A 00 01 05	\\...M.....
00000007	65 4A 00 00 08 00 00 00 76 4E A7 59 78 8A DA 01	eJ.....vNSYxSÚ.
00000008	5A 62 02 00 00 00 00 00 05 00 00 00 01 00 00 00	Zb.....
00000009	01 00 00 00 08 00 00 00 00 00 00 00 00 09 00 00
0000000A	09 00 00 00 00 00 00 00 06 00 00 00 00 00 00 00
0000000B	E4 FD FF FF 40 00 74 00 7A 00 72 00 65 00 73 00	äýýý@.t.z.r.e.s.
0000000C	2E 00 64 00 6C 00 6C 00 2C 00 2D 00 36 00 33 00	..d.l.l.,.-.6.3.
0000000D	32 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	2.....
0000000E	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000F	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000010	00 00 00 00 00 00 00 00 40 00 74 00 7A 00 72 00@.t.z.r.
00000011	65 00 73 00 2E 00 64 00 6C 00 6C 00 2C 00 2D 00	e.s...d.l.l.,.-.
00000012	36 00 33 00 31 00 00 00 00 00 00 00 00 00 00 00	6.3.l.....
00000013	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000014	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000015	00 00 00 00 00 00 00 00 FF FF FF FF 00 00 00 00Äýýý
00000016	40 9F DE 9F 00 7A 01 80 96 98 00 00 00 00 00 00	@Yb".ú.€-.....
00000017	63 0D 2F C9 77 8A DA 01 01 00 00 00 00 00 00 00	c./ÉwSÚ.....
00000018	49 00 6E 00 63 00 69 00 64 00 65 00 6E 00 74 00	I.n.c.i.d.e.n.t.
00000019	20 00 52 00 65 00 73 00 70 00 6F 00 6E 00 73 00	.R.e.s.p.o.n.s.
0000001A	65 00 00 00 43 00 3A 00 5C 00 55 00 73 00 65 00	e...C.:.\.U.s.e.
0000001B	72 00 73 00 5C 00 6B 00 61 00 6E 00 72 00 69 00	r.s.\.k.a.n.r.i.
0000001C	5C 00 41 00 70 00 70 00 44 00 61 00 74 00 61 00	\.A.p.p.D.a.t.a.
0000001D	5C 00 4C 00 6F 00 63 00 61 00 6C 00 5C 00 49 00	\.L.o.c.a.l.\.I.
0000001E	6E 00 63 00 69 00 64 00 65 00 6E 00 74 00 20 00	n.c.i.d.e.n.t. .
0000001F	52 00 65 00 73 00 70 00 6F 00 6E 00 73 00 65 00	R.e.s.p.o.n.s.e.
00000200	2E 00 65 00 74 00 6C 00 00 00 00 00 00 00 00 00	..e.t.l.....
00000201	02 00 02 C0 50 00 50 00 44 3B 00 00 18 29 00 00	..ÄP.P.D;...)
00000202	95 CF 9F 3D 71 0B 00 00 5C 08 00 00 4D 04 00 00	*iY=q...\..M...
00000203	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000204	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000205	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000206	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
00000207	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy

ETL File Header

_WMI_BUFFER_HEADER

- ❑ ETL file format is **undocumented**.
- ❑ ETL header is defined in the **WMI_BUFFER_HEADER** structure in the Windows symbol.
- ❑ No signature.

```
//0x48 bytes (sizeof)
struct _WMI_BUFFER_HEADER
{
    ULONG BufferSize;
    ULONG SavedOffset;
    volatile ULONG CurrentOffset;
    volatile LONG ReferenceCount;
    union _LARGE_INTEGER TimeStamp;
    LONGLONG SequenceNumber;
    union
    {
        struct
        {
            ULONGLONG ClockType:3;
            ULONGLONG Frequency:61;
        };
        struct _SINGLE_LIST_ENTRY SlistEntry;
        struct _WMI_BUFFER_HEADER* NextBuffer;
    };
    struct _ETW_BUFFER_CONTEXT ClientContext;
    enum _ETW_BUFFER_STATE State;
    ULONG Offset;
    USHORT BufferFlag;
    USHORT BufferType;
    union
    {
        ULONG Padding1[4];
        struct _ETW_REF_CLOCK ReferenceTime;
        struct _LIST_ENTRY GlobalEntry;
        struct
        {
            VOID* Pointer0;
            VOID* Pointer1;
        };
    };
};
};
```


ETL File Format

Trace Header

```
00000000 00 20 00 00 60 02 00 00 60 02 00 00 00 00 00 00 .....
00000001 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000002 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....&.....
00000003 60 02 00 00 21 00 04 00 00 00 00 00 00 00 00 00 .....!.....
00000004 00 00 00 00 00 00 00 00 02 00 00 00 C2 01 00 00 .....AA...
00000005 44 3B 00 00 18 29 00 00 95 CF 5D 71 0B 00 00 D;...)..*ÿ=q...
00000006 5C 08 00 00 4D 04 00 00 00 2F 00 0A 00 01 05 \\.M.....
00000007 65 4A 00 00 08 00 00 00 76 00 0A DA 01 eJ.....vNSYxSÚ.
00000008 5A 62 02 00 00 00 00 00 00 00 00 00 00 00 .....
00000009 01 00 00 00 08 00 00 00 00 00 00 00 00 00 .....
0000000A 09 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0000000B E4 FD FF FF 40 00 00 00 00 00 00 00 00 00 .....
0000000C 2E 00 64 00 6C 00 00 00 00 00 00 00 00 00 .....
0000000D 32 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0000000E 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0000000F 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000010 00 00 00 00 00 00 00 00 40 00 74 00 7A 00 72 00 .....@.t.z.r.
00000011 65 00 73 00 2E 00 64 00 6C 00 6C 00 2C 00 2D 00 e.s...d.l.l.,-..
00000012 36 00 33 00 31 00 00 00 00 00 00 00 00 00 00 00 6.3.l.....
00000013 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000014 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000015 00 00 00 00 00 00 00 00 C4 FF FF FF 00 00 00 00 .....Äÿÿÿ...
00000016 40 9F DE 94 06 7F DA 01 80 96 98 00 00 00 00 00 00 @ÿ".ú.€-.....
00000017 63 0D 2F C9 77 8A DA 01 01 00 00 00 00 00 00 00 00 c./ÉwSÚ.....
00000018 49 00 6E 00 63 00 69 00 64 00 65 00 6E 00 74 00 I.n.c.i.d.e.n.t.
00000019 20 00 52 00 65 00 73 00 70 00 6F 00 6E 00 73 00 .R.e.s.p.o.n.s.
0000001A 65 00 00 00 43 00 3A 00 5C 00 55 00 73 00 65 00 e...C.:.\.U.s.e.
0000001B 72 00 73 00 5C 00 6B 00 61 00 6E 00 72 00 69 00 r.s.\.k.a.n.r.i.
0000001C 5C 00 41 00 70 00 70 00 44 00 61 00 74 00 61 00 \.A.p.p.D.a.t.a.
0000001D 5C 00 4C 00 6F 00 63 00 61 00 6C 00 5C 00 49 00 \.L.o.c.a.l.\.I.
0000001E 6E 00 63 00 69 00 64 00 65 00 6E 00 74 00 20 00 n.c.i.d.e.n.t..
0000001F 52 00 65 00 73 00 70 00 6F 00 6E 00 73 00 65 00 R.e.s.p.o.n.s.e.
00000200 2E 00 65 00 74 00 6C 00 00 00 00 00 00 00 00 00 ..e.t.l.....
00000201 02 00 02 C0 50 00 50 00 44 3B 00 00 18 29 00 00 ...ÄP.P.D;...)..
00000202 95 CF 9F 3D 71 0B 00 00 5C 08 00 00 4D 04 00 00 *ÿÿ=q...\..M...
00000203 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000204 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000205 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000206 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF yyyyyyyyyyyyyyyy
00000207 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF yyyyyyyyyyyyyyyy
```

0x02: Header Type

Trace Header

```
HeaderType {  
    TRACE_HEADER_TYPE_SYSTEM32           = 1,  
    TRACE_HEADER_TYPE_SYSTEM64          = 2,  
    TRACE_HEADER_TYPE_COMPACT32         = 3,  
    TRACE_HEADER_TYPE_COMPACT64         = 4,  
    TRACE_HEADER_TYPE_FULL_HEADER32     = 10,  
    TRACE_HEADER_TYPE_INSTANCE32        = 11,  
    TRACE_HEADER_TYPE_TIMED              = 12,  
    TRACE_HEADER_TYPE_ERROR              = 13,  
    TRACE_HEADER_TYPE_WNODE_HEADER      = 14,  
    TRACE_HEADER_TYPE_MESSAGE           = 15,  
    TRACE_HEADER_TYPE_PERFINF032        = 16,  
    TRACE_HEADER_TYPE_PERFINF064        = 17,  
    TRACE_HEADER_TYPE_EVENT_HEADER32    = 18,  
    TRACE_HEADER_TYPE_EVENT_HEADER64    = 19,  
    TRACE_HEADER_TYPE_FULL_HEADER64     = 20,  
    TRACE_HEADER_TYPE_INSTANCE64        = 21  
};
```

Trace Header

```
HeaderType {  
    TRACE_HEADER_TYPE_SYSTEM32      = 1,  
    TRACE_HEADER_TYPE_SYSTEM64     = 2,  
    TRACE_HEADER_TYPE_COMPACT32     = 3,  
    TRACE_HEADER_TYPE_COMPACT64     = 4,  
    TRACE_HEADER_TYPE_FULL_HEADER32 = 10,  
    TRACE_HEADER_TYPE_INSTANCE32    = 11,  
    TRACE_HEADER_TYPE_TIMED         = 12,  
    TRACE_HEADER_TYPE_ERROR         = 13,  
    TRACE_HEADER_TYPE_WNODE_HEADER  = 14,  
    TRACE_HEADER_TYPE_MESSAGE       = 15,  
    TRACE_HEADER_TYPE_PERFINF032    = 16,  
    TRACE_HEADER_TYPE_PERFINF064    = 17,  
    TRACE_HEADER_TYPE_EVENT_HEADER32 = 18,  
    TRACE_HEADER_TYPE_EVENT_HEADER64 = 19,  
    TRACE_HEADER_TYPE_FULL_HEADER64 = 20,  
    TRACE_HEADER_TYPE_INSTANCE64    = 21  
};
```

```
//0x20 bytes (sizeof)  
struct _SYSTEM_TRACE_HEADER  
{  
    union  
    {  
        ULONG Marker;  
        struct  
        {  
            USHORT Version;  
            UCHAR HeaderType;  
            UCHAR Flags;  
        };  
    };  
    union  
    {  
        ULONG Header;  
        struct _WMI_TRACE_PACKET Packet;  
    };  
    ULONG ThreadId;  
    ULONG ProcessId;  
    union _LARGE_INTEGER SystemTime;  
    ULONG KernelTime;  
    ULONG UserTime;  
};
```

Trace Header

```
HeaderType {  
    TRACE_HEADER_TYPE_SYSTEM32           = 1,  
    TRACE_HEADER_TYPE_SYSTEM64           = 2,  
    TRACE_HEADER_TYPE_COMPACT32          = 3,  
    TRACE_HEADER_TYPE_COMPACT64          = 4,  
    TRACE_HEADER_TYPE_FULL_HEADER32      = 10,  
    TRACE_HEADER_TYPE_INSTANCE32         = 11,  
    TRACE_HEADER_TYPE_TIMED               = 12,  
    TRACE_HEADER_TYPE_ERROR               = 13,  
    TRACE_HEADER_TYPE_WNODE_HEADER        = 14,  
    TRACE_HEADER_TYPE_MESSAGE              = 15,  
    TRACE_HEADER_TYPE_PERFINF032          = 16,  
    TRACE_HEADER_TYPE_PERFINF064          = 17,  
    TRACE_HEADER_TYPE_EVENT_HEADER32      = 18,  
    TRACE_HEADER_TYPE_EVENT_HEADER64     = 19,  
    TRACE_HEADER_TYPE_FULL_HEADER64      = 20,  
    TRACE_HEADER_TYPE_INSTANCE64         = 21  
};
```

```
//0x50 bytes (sizeof)  
struct _EVENT_HEADER  
{  
    USHORT Size;  
    USHORT HeaderType;  
    USHORT Flags;  
    USHORT EventProperty;  
    ULONG ThreadId;  
    ULONG ProcessId;  
    union _LARGE_INTEGER TimeStamp;  
    struct _GUID ProviderId;  
    struct _EVENT_DESCRIPTOR EventDescriptor;  
    union  
    {  
        struct  
        {  
            ULONG KernelTime;  
            ULONG UserTime;  
        };  
        ULONGLONG ProcessorTime;  
    };  
    struct _GUID ActivityId;  
};
```


_EVENT_HEADER

```
Flags {  
    EVENT_HEADER_FLAG_EXTENDED_INFO = 0x01,  
    EVENT_HEADER_FLAG_PRIVATE_SESSION = 0x02,  
    EVENT_HEADER_FLAG_STRING_ONLY = 0x04,  
    EVENT_HEADER_FLAG_TRACE_MESSAGE = 0x08,  
    EVENT_HEADER_FLAG_NO_CPUTIME = 0x10,  
    EVENT_HEADER_FLAG_32_BIT_HEADER = 0x20,  
    EVENT_HEADER_FLAG_64_BIT_HEADER = 0x40,  
    EVENT_HEADER_FLAG_CLASSIC_HEADER = 0x100,  
    EVENT_HEADER_FLAG_PROCESSOR_INDEX = 0x200,  
};
```



EVENT_HEADER_FLAG_EXTENDED_INFO adds 8 bytes of header after **_EVENT_HEADER**.

__EVENT_HEADER

```
00010000 00 00 01 00 D0 25 00 00 D0 25 01 00 00 00 00 00 00 00 .....D%.D%.....
00010001 00 99 00 00 00 00 00 00 01 00 00 00 00 00 00 00 ..%.Ö.....
00010002 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....).....
00010003 D0 25 00 00 21 00 00 00 38 30 72 67 83 83 FF FF D%.!.!..80rgffÿÿ
00010040 20 C9 8F 62 83 83 FF FF 06 01 13 C0 01 00 00 00 00 00 È.bffÿÿ...Ä....
00010050 CC 06 00 00 7C 45 00 00 1F CA EF 12 D2 0F 00 00 00 00 i...|E...Èi.Ö...
00010060 D0 25 00 00 00 00 00 00 00 00 00 00 00 00 00 00 BÄÄ-U9.A³0....æk
00010070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00010080 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00010090 00 00 00 00 00 00 00 00 48 00 0C 00 01 00 39 00 H....9.
000100A0 39 00 4D 69 63 72 6F 73 6F 66 74 2E 57 69 6E 64 9.M...
000100B0 6F 77 73 2E 55 70 64 61 74 65 48 65 61 6C 74 68 .....
000100C0 54 6F 6F 6C 73 00 13 00 01 1A 73 50 4F CF 89 82 .....
000100D0 47 B3 E0 DC E8 C9 04 76 BA 00 00 00 00 00 00 00 00 .....
000100E0 30 00 0B 00 00 00 28 00 28 00 00 49 6E 66 6F 72 .....
000100F0 6D 61 74 69 6F 6E 00 50 61 63 6B 61 67 65 56 65 .....
00010100 72 73 69 6F 6E 00 02 4D 65 73 73 61 67 65 00 01 .....
00010110 32 30 32 33 2E 31 30 00 53 00 74 00 61 00 72 00 .....
00010120 74 00 65 00 64 00 20 00 49 00 6E 00 73 00 74 00 .....
00010130 61 00 6C 00 6C 00 48 00 65 00 61 00 6C 00 74 00 .....
00010140 68 00 54 00 6F 00 6F 00 6C 00 73 00 00 00 00 00 .....
00010150 81 04 13 C0 01 00 00 00 CC 06 00 00 7C 45 00 00 .....
00010160 76 8F F1 12 D2 0F 00 00 DE C2 C1 B7 55 39 03 41 vžñ.Ö...BÄÄ-U9.A
00010170 B3 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 *0....æk.....
00010180 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00010190 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000101A0 48 00 0C 00 01 00 39 00 39 00 4D 69 63 72 6F 73 H....9.9.Micros
000101B0 6F 66 74 2E 57 69 6E 64 6F 77 73 2E 55 70 64 61 oft.Windows.Upda
000101C0 74 65 48 65 61 6C 74 68 54 6F 6F 6C 73 00 13 00 teHealthTools...
000101D0 01 1A 73 50 4F CF 89 82 47 B3 E0 DC E8 C9 04 76 ..sPÖIt,G'aÜeÉ.v
000101E0 BA 00 00 00 00 00 00 00 88 03 0B 00 00 00 79 03 °.....^.....y.
000101F0 79 03 8A 20 55 6E 69 66 69 65 64 49 6E 73 74 61 y.Š UnifiedInsta
00010200 6C 6C 65 72 53 74 61 72 74 00 50 61 72 74 41 5F llerStart..PartA_
00010210 50 72 69 76 54 61 67 73 00 0A 50 61 63 6B 61 67 PrivTags..Packag
00010220 65 56 65 72 73 69 6F 6E 00 02 47 6C 6F 62 61 6C eVersion..Global
00010230 45 76 65 6E 74 43 6F 75 6E 74 65 72 00 0A 43 56 EventCounter..CV
00010240 00 02 55 6E 69 66 69 65 64 49 6E 73 74 61 6C 6C ..UnifiedInstall
00010250 65 72 50 6C 61 74 66 6F 72 6D 52 65 73 75 6C 74 erPlatformResult
00010260 00 07 55 6E 69 66 69 65 64 49 6E 73 74 61 6C 6C ..UnifiedInstall
00010270 65 72 50 6C 61 74 66 6F 72 6D 54 79 70 65 00 08 erPlatformType..
```

WMI_BUFFER_HEADER

EVENT_HEADER

EVENT_HEADER

EVENT_HEADER_EXT
ENDED_DATA_ITEM

__EVENT_HEADER

```
00010000 00 00 01 00 D0 25 00 00 D0 25 01 00 00 00 00 00 .....D%.D%.....
00010001 00 99 00 00 00 00 00 00 01 00 00 00 00 00 .....%6.Ö.....
00010002 00 00 00 00 00 00 00 00 00 00 00 00 00 .....).....
00010030 D0 25 00 00 21 00 00 00 38 30 72 67 83 83 FF FF D%.!...80rgffÿÿ
00010040 20 C9 8F 62 83 83 FF FF 06 01 13 C0 01 00 00 00 Ę.bffÿÿ...Ä....
00010050 CC 06 00 00 7C 45 00 00 1F CA EF 12 D2 0F 00 00 ĩ...E...Ë.Ö....
00010060 D1 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00010070 01 00 01 00 00 00 00 00 00 00 00 00 00 00 .....
00010080 01 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00010090 00 00 00 00 00 00 00 00 48 00 0C 00 .....
000100A0 39 00 4D 69 63 72 6F 73 6F 66 74 2E 57 69 68
000100B0 6F 77 73 2E 55 70 64 61 74 65 48 65 61 6C 74 68 ows.UpdateHealth
000100C0 54 6F 6F 6C 73 00 13 00 01 1A 73 50 4F CF 89 82 Tools.....sPÖÏ%,
000100D0 47 B3 E0 DC E8 C9 04 76 BA 00 00 00 00 00 00 00 G'aÜëË.v^.....
000100E0 30 00 0B 00 00 00 28 00 28 00 00 49 6E 66 6F 72 0.....(.Infor
000100F0 6D 61 74 69 6F 6E 00 50 61 63 6B 61 67 65 56 65 mation.PackageVe
00010100 72 73 69 6F 6E 00 02 4D 65 73 73 61 67 65 00 01 rsion..Message..
00010110 32 30 32 33 2E 31 30 00 53 00 74 00 61 00 72 00 2023.10.S.t.a.r.
00010120 74 00 65 00 64 00 20 00 49 00 6E 00 73 00 74 00 t.e.d..I.n.s.t.
00010130 61 00 6C 00 6C 00 48 00 65 00 61 00 6C 00 74 00 a.l.l.H.e.a.l.t.
00010140 68 00 54 00 6F 00 6F 00 6C 00 73 00 00 00 00 00 h.T.o.o.l.s.....
00010150 81 04 13 C0 01 00 00 00 CC 06 00 00 7C 45 00 00 ...Ä...i...|E..
00010160 76 8F F1 12 D2 0F 00 00 DE C2 C1 B7 55 39 03 41 vžñ.Ö...PÄÄ-U9.A
00010170 B3 00 00 00 00 00 00 00 00 00 00 00 00 00 00 *0....#k.....
00010180 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....ë.....
00010190 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000101A0 48 00 0C 00 01 00 39 00 39 00 4D 69 63 72 6F 73 H.....9.9.Micros
000101B0 6F 66 74 2E 57 69 6E 64 6F 77 73 2E 55 70 64 61 oft.Windows.Upda
000101C0 74 65 48 65 61 6C 74 68 54 6F 6F 6C 73 00 13 00 teHealthTools...
000101D0 01 1A 73 50 4F CF 89 82 47 B3 E0 DC E8 C9 04 76 ..sPÖÏ%,G'aÜëË.v
000101E0 BA 00 00 00 00 00 00 00 88 03 0B 00 00 00 79 03 ^.....^.....y.
000101F0 79 03 8A 20 55 6E 69 66 69 65 64 49 6E 73 74 61 y.Š UnifiedInsta
00010200 6C 6C 65 72 53 74 61 72 74 00 50 61 72 74 41 5F llerStart..PartA_
00010210 50 72 69 76 54 61 67 73 00 0A 50 61 63 6B 61 67 PrivTags..Packag
00010220 65 56 65 72 73 69 6F 6E 00 02 47 6C 6F 62 61 6C eVersion..Global
00010230 45 76 65 6E 74 43 6F 75 6E 74 65 72 00 0A 43 56 EventCounter..CV
00010240 00 02 55 6E 69 66 69 65 64 49 6E 73 74 61 6C 6C ..UnifiedInstall
00010250 65 72 50 6C 61 74 66 6F 72 6D 52 65 73 75 6C 74 erPlatformResult
00010260 00 07 55 6E 69 66 69 65 64 49 6E 73 74 61 6C 6C ..UnifiedInstall
00010270 65 72 50 6C 61 74 66 6F 72 6D 54 79 70 65 00 08 erPlatformType..
```

WMI_BUFFER_HEADER

__EVENT_HEADER

__EVENT_HEADER

0x000C: ExtType

__EVENT_HEADER_EXTENDED_DATA_ITEM

```
//0x10 bytes (sizeof)
struct __EVENT_HEADER_EXTENDED_DATA_ITEM
{
    USHORT Reserved1;
    USHORT ExtType;
    USHORT Linkage:1;
    USHORT Reserved2:15;
    USHORT DataSize;
    ULONGLONG DataPtr;
};
```

```
ExtType {
    EVENT_HEADER_EXT_TYPE_RELATED_ACTIVITYID = 0x0001,
    EVENT_HEADER_EXT_TYPE_SID                = 0x0002,
    EVENT_HEADER_EXT_TYPE_TS_ID              = 0x0003,
    EVENT_HEADER_EXT_TYPE_INSTANCE_INFO     = 0x0004,
    EVENT_HEADER_EXT_TYPE_STACK_TRACE32     = 0x0005,
    EVENT_HEADER_EXT_TYPE_STACK_TRACE64     = 0x0006,
    EVENT_HEADER_EXT_TYPE_PEBS_INDEX        = 0x0007,
    EVENT_HEADER_EXT_TYPE_PMC_COUNTERS      = 0x0008,
    EVENT_HEADER_EXT_TYPE_PSM_KEY           = 0x0009,
    EVENT_HEADER_EXT_TYPE_EVENT_KEY         = 0x000A,
    EVENT_HEADER_EXT_TYPE_EVENT_SCHEMA_TL   = 0x000B,
    EVENT_HEADER_EXT_TYPE_PROV_TRAITS       = 0x000C,
    EVENT_HEADER_EXT_TYPE_PROCESS_START_KEY = 0x000D,
    EVENT_HEADER_EXT_TYPE_MAX               = 0x000E,
};
```

__EVENT_HEADER_EXTENDED_DATA_ITEM

```
//0x10 bytes (sizeof)
struct __EVENT_HEADER_EXTENDED_DATA_ITEM
{
    USHORT Reserved1;
    USHORT ExtType;
    USHORT Linkage:1;
    USHORT Reserved2:15;
    USHORT DataSize;
    ULONGLONG DataPtr;
};
```

```
ExtType {
    EVENT_HEADER_EXT_TYPE_RELATED_ACTIVITYID = 0x0001,
    EVENT_HEADER_EXT_TYPE_SID                = 0x0002,
    EVENT_HEADER_EXT_TYPE_TS_ID              = 0x0003,
    EVENT_HEADER_EXT_TYPE_INSTANCE_INFO     = 0x0004,
    EVENT_HEADER_EXT_TYPE_STACK_TRACE32     = 0x0005,
    EVENT_HEADER_EXT_TYPE_STACK_TRACE64     = 0x0006,
    EVENT_HEADER_EXT_TYPE_PEBS_INDEX        = 0x0007,
    EVENT_HEADER_EXT_TYPE_PMC_COUNTERS      = 0x0008,
    EVENT_HEADER_EXT_TYPE_PSM_KEY           = 0x0009,
    EVENT_HEADER_EXT_TYPE_EVENT_KEY         = 0x000A,
    EVENT_HEADER_EXT_TYPE_EVENT_SCHEMA_TL   = 0x000B,
    EVENT_HEADER_EXT_TYPE_PROV_TRAITS       = 0x000C,
    EVENT_HEADER_EXT_TYPE_PROCESS_START_KEY = 0x000D,
    EVENT_HEADER_EXT_TYPE_M                 = 0x000E,
};
```

byte array

__EVENT_HEADER

00010000	00 00 01 00 D0 25 00 00 D0 25 01 00 00 00 00 00D%.D%.....
00010001	99 00 00 00 00 00 00 00 01 00 00 00 00 00 00	.%6.Ö.....
00010002	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00)
00010030	D0 25 00 00 21 00 00 00 38 30 72 67 83 83 FF FF	D%.!...80rgffÿÿ
00010040	20 C9 8F 62 83 83 FF FF 06 01 13 C0 01 00 00 00	Ë.bffÿÿ...Ä....
00010050	CC 06 00 00 7C 45 00 00 1F CA EF 12 D2 0F 00 00	ì... E...È.ì.Ö...
00010060	D0 25 00 00 00 00 00 00 00 00 00 00 00 00 00	ÈÄÄ-U9.A³0....æk
00010070	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00010080	01 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00010090	00 00 00 00 00 00 00 00 48 00 0C 00 01 00 39 00H.....9.
000100A0	39 00 4D 69 63 72 6F 73 6F 66 74 2E 57 69 6E 64	9.Microsoft.Wind
000100B0	6F 77 73 2E 55 70 64 61 74 65 48 65 61 6C 74 68	ows.UpdateHealth
000100C0	54 6F 6F 6C 73 00 13 00 01 1A 73 50 4F CF 89 82	Tools.....sPOi%,
000100D0	47 B3 E0 DC E8 C9 04 76 BA 00 00 00 00 00 00 00	G'äÜëÉ.v°.....
000100E0	30 00 0B 00 00 00 38 00 28 00 00 49 6E 66 6F 72	0.....(.(.Infor
00010100	5D 61 74 40 6E 02 4D 65 73 73 61 67 65 56 65	mation.PackageVe
00010110	72 73 69 64 6E 02 4D 65 73 73 61 67 65 00 01	rsion..Message..
00010120	32 30 32 33 2E 31 30 00 53 00 74 00 61 00 72 00	2023.10.S.t.a.r.
00010130	74 00 65 00 64 00 20 00 49 00 6E 00 73 00 74 00	t.e.d. .I.n.s.t.
00010140	61 00 6C 00 6C 00 48 00 65 00 61 00 6C 00 74 00	a.l.l.H.e.a.l.t.
00010150	68 00 54 00 6F 00 6F 00 6C 00 73 00 00 00 00 00	h.T.o.o.l.s.....
00010160	81 04 13 C0 01 00 00 00 CC 06 00 00 7C 45 00 00	...Ä...ì... E..
00010170	76 8F F1 12 D2 0F 00 00 DE C2 C1 B7 55 39 03 41	vžň.Ö...ÈÄÄ-U9.A
00010180	B3 00 00 00 00 00 00 00 00 00 00 00 00 00 00	³0....æk.....
00010190	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00È.....
000101A0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000101B0	48 00 0C 00 01 00 39 00 39 00 4D 69 63 72 6F 73	H.....9.9.Micros
000101C0	6F 66 74 2E 57 69 6E 64 6F 77 73 2E 55 70 64 61	oft.Windows.Upda
000101D0	74 65 48 65 61 6C 74 68 54 6F 6F 6C 73 00 13 00	teHealthTools...
000101E0	01 1A 73 50 4F CF 89 82 47 B3 E0 DC E8 C9 04 76	..sPOi%,G'äÜëÉ.v
000101F0	BA 00 00 00 00 00 00 00 88 03 0B 00 00 00 79 03	°.....^.....y.
00010200	79 03 8A 20 55 6E 69 66 69 65 64 49 6E 73 74 61	y.Š UnifiedInsta
00010210	6C 6C 65 72 53 74 51 72 74 00 50 61 72 74 41 5F	llerStart..PartA_
00010220	50 72 69 70 0A 50 61 63 6B 61 67 65 56 65 72 73	PrivTags..Packag
00010230	65 56 65 72 73 59 6F 6E 00 02 47 6C 6F 62 61 6C	eVersion..Global
00010240	45 76 65 6E 74 43 6F 75 6E 74 65 72 00 0A 43 56	EventCounter..CV
00010250	00 02 55 6E 69 66 69 65 64 49 6E 73 74 61 6C 6C	..UnifiedInstall
00010260	65 72 50 6C 61 74 66 6F 72 6D 52 65 73 75 6C 74	erPlatformResult
00010270	00 07 55 6E 69 66 69 65 64 49 6E 73 74 61 6C 6C	..UnifiedInstall
00010280	65 72 50 6C 61 74 66 6F 72 6D 54 79 70 65 00 08	erPlatformType..

Byte array

payload

payload

Tracing ETW from Structures

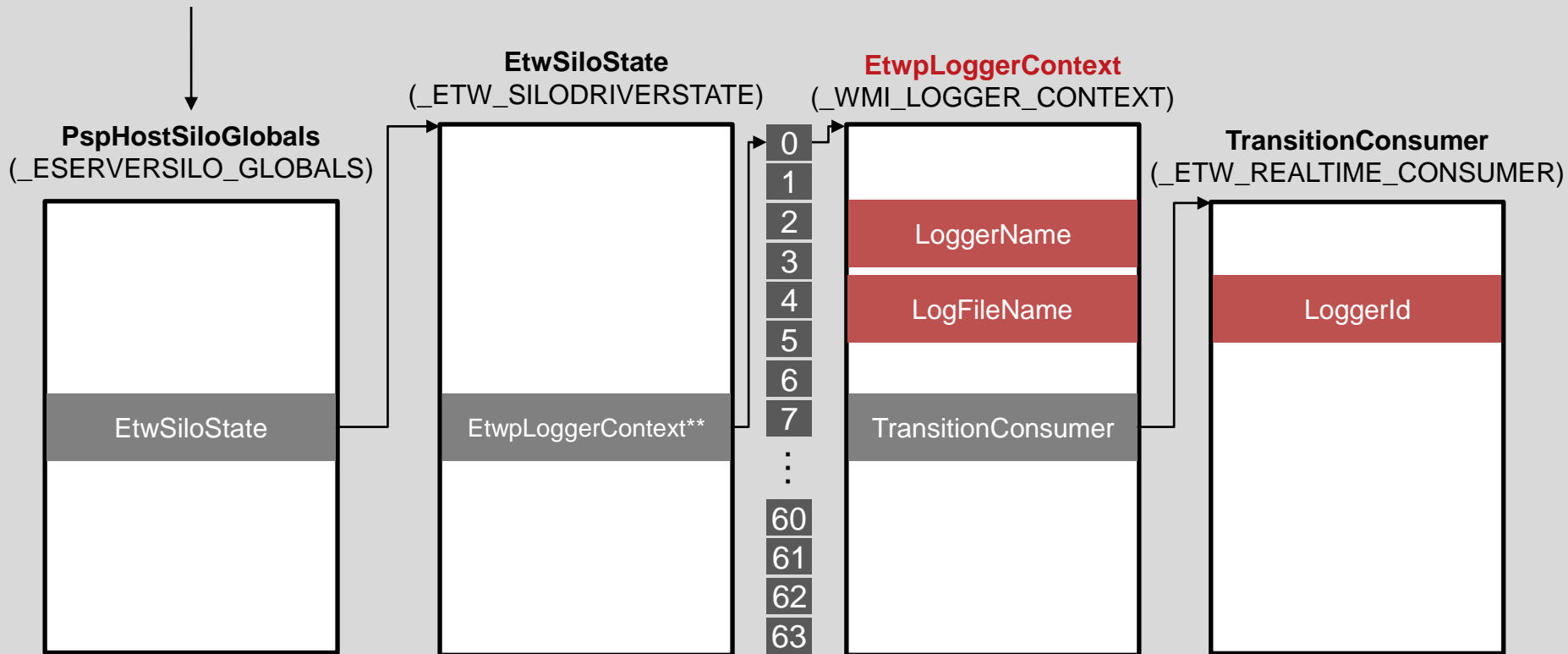
ETW data can be traced from **PspHostSiloGlobals** structure.



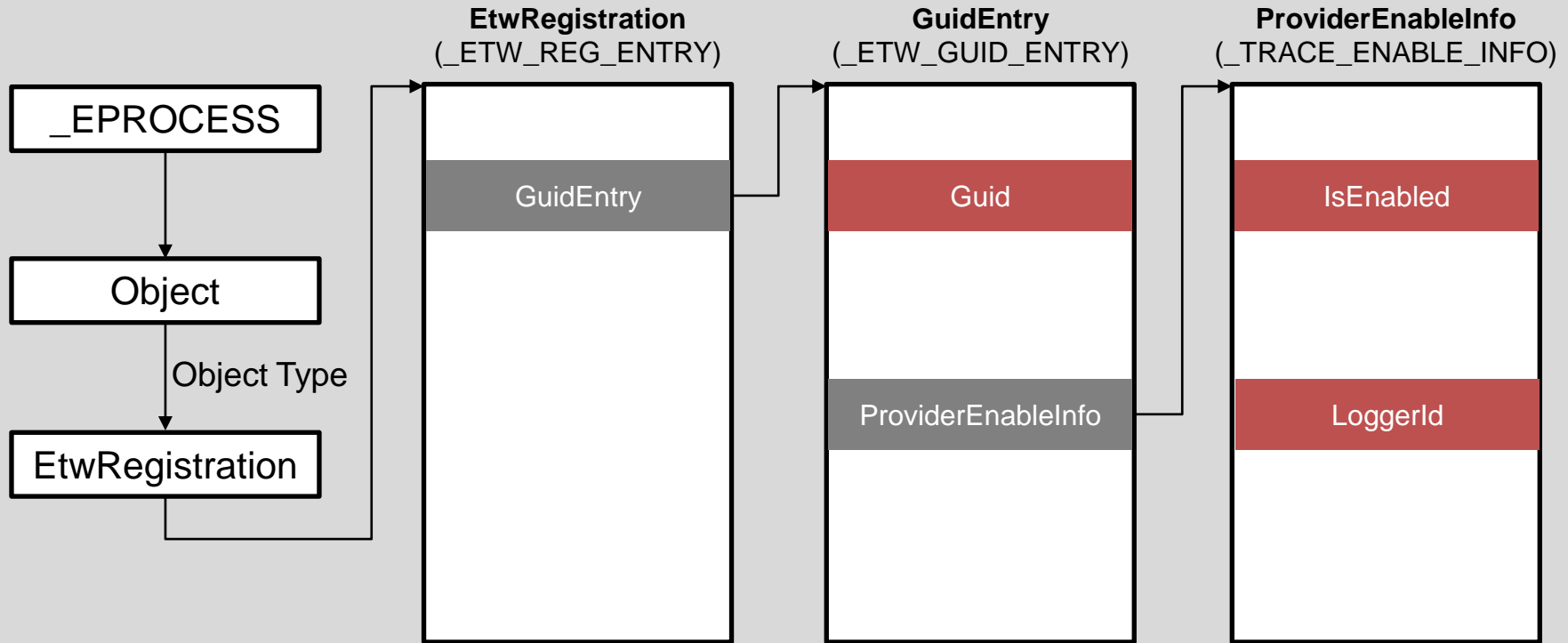
PspHostSiloGlobals structure can be retrieved from the **PsGetCurrentServerSiloGlobals()** Windows API.

How to Get ETW Consumers from Structures

PsGetCurrentServerSiloGlobals()



How to Get ETW Providers from Structures

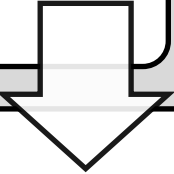


ETW Data Structure

It is not possible to list all ETW Providers or get detailed ETW settings from the **Performance Monitor**.



For detailed information on ETW, it is necessary to trace the structure in **kernel mode**.



Access kernel mode from memory image to trace ETW.

Tracing ETW from Memory

Tracing ETW Providers using Volatility

```
ir-mbp:volatility3 tomonaga$ python3 vol.py -f test.mem -p plugin etwscan.etwProvider
Volatility 3 Framework 2.7.0
Progress: 100.00 PDB scanning finished
PID ImageFileName type_map address guid LoggerId Level EnableMask
316 smss.exe EtwRegistration 0xe5861227f160 43e63da5-41d1-4fbf-aded-1bbed98fdd1d 0 No 00000001
408 csrss.exe EtwRegistration 0xe58612ffbf60 e8316a2d-0d94-4f52-85dd-1e15b66c5891 0 TRACE_LEVEL_INFORMA
TION 00000001
408 csrss.exe EtwRegistration 0xe58612ffbcc0 f1ef270a-0d32-4352-ba52-dbab41e1d859 0 No 00000001
408 csrss.exe EtwRegistration 0xe586147f2400 9d55b53d-449b-4824-a637-24f9d69aa02f 0 No 00000001
408 csrss.exe EtwRegistration 0xe586171a0a30 f4aed7c7-a898-4627-b053-44a7caa12fcd 0 No 00000001
480 wininit.exe EtwRegistration 0xe586147f2e80 f1ef270a-0d32-4352-ba52-dbab41e1d859 0 No 00000001
480 wininit.exe EtwRegistration 0xe58614708860 206f6dea-d3c5-4d10-bc72-989f03c8b84b 0 No 00000111
480 wininit.exe EtwRegistration 0xe586147084e0 f4aed7c7-a898-4627-b053-44a7caa12fcd 0 No 00000001
480 wininit.exe EtwRegistration 0xe586147f2160 db00dfb6-29f9-4a9c-9b3b-1f4f9e7d9770 0 No 00000001
480 wininit.exe EtwRegistration 0xe586147f1b40 fc65ddd8-d6ef-4962-83d5-6e5cfe9ce148 0 No 00000111
480 wininit.exe EtwRegistration 0xe5861470d6e0 16a1adc1-9b7f-4cd9-94b3-d8296ab1b130 0 No 00000001
480 wininit.exe EtwRegistration 0xe58614efe2a0 db00dfb6-29f9-4a9c-9b3b-1f4f9e7d9770 0 No 00000001
480 wininit.exe EtwRegistration 0xe58614efd120 f1ef270a-0d32-4352-ba52-dbab41e1d859 0 No 00000001
480 wininit.exe EtwRegistration 0xe58615025240 1c95126e-7eea-49a9-a3fe-a378b03ddb4d 0 No 00000011
576 services.exe EtwRegistration 0xe5861456eb40 f1ef270a-0d32-4352-ba52-dbab41e1d859 0 No 00000001
576 services.exe EtwRegistration 0xe5861456fcc0 555908d1-a6d7-4695-8e1e-26931d2012f4 0 No 00000001
576 services.exe EtwRegistration 0xe58614ea6240 0063715b-eeda-4007-9429-ad526f62696e 0 TRACE_LEVEL_INFORMA
TION 00000001
576 services.exe EtwRegistration 0xe58614efbde0 f4aed7c7-a898-4627-b053-44a7caa12fcd 0 No 00000001
576 services.exe EtwRegistration 0xe58614efc400 2e35aaeb-857f-4beb-a418-2e6c0e54d988 0 No 00000001
```


What is Volatility Framework?



The Volatility Framework was developed as an **open-source memory forensics tool** written in Python.

This tool is the **de facto standard** for memory forensics.

How to Use

```
$ git clone https://github.com/volatilityfoundation/volatility3.git
$ cd volatility3
$ pip3 install -r requirements.txt
$ python3 vol.py -f test.mem windows.info
```

<https://volatilityfoundation.org/>

Tracing ETW from Memory

Tracing ETW Providers using Volatility

```
ir-mbp:volatility3 tomonaga$ python3 vol.py -f test.mem -p plugin etwscan.etwProvider
Volatility 3 Framework 2.7.0
Progress: 100.00          PDB scanning finished
PID      ImageFileName      type_map      address guid      LoggerId      Level      EnableMask
316      smss.exe           EtwRegistration 0xe5861227f160    43e63da5-41d1-4fbf-aded-1bbed98fdd1d    0      No      00000001
408      csrss.exe         EtwRegistration 0xe58612ffbf60    e8316a2d-0d94-4f52-85dd-1e15b66c5891    0      TRACE_LEVEL_INFORMA
TION     00000001
408      ...
408      ...
408      ...
480      wininit.exe       EtwRegistration 0xe586147f1b40    fc65ddd8-d6ef-4962-83d5-6e5cfe9ce148    0      No      00000111
480      wininit.exe       EtwRegistration 0xe5861470d6e0    16a1adc1-9b7f-4cd9-94b3-d8296ab1b130    0      No      00000001
480      wininit.exe       EtwRegistration 0xe58614efe2a0    db00dfb6-29f9-4a9c-9b3b-1f4f9e7d9770    0      No      00000001
480      wininit.exe       EtwRegistration 0xe58614efd120    f1ef270a-0d32-4352-ba52-dbab41e1d859    0      No      00000001
480      wininit.exe       EtwRegistration 0xe58615025240    1c95126e-7eea-49a9-a3fe-a378b03ddb4d    0      No      00000011
576      services.exe      EtwRegistration 0xe5861456eb40    f1ef270a-0d32-4352-ba52-dbab41e1d859    0      No      00000001
576      services.exe      EtwRegistration 0xe5861456fcc0    555908d1-a6d7-4695-8e1e-26931d2012f4    0      No      00000001
576      services.exe      EtwRegistration 0xe58614ea6240    0063715b-eeda-4007-9429-ad526f62696e    0      TRACE_LEVEL_INFORMA
TION     00000001
576      services.exe      EtwRegistration 0xe58614efbde0    f4aed7c7-a898-4627-b053-44a7caa12fcd    0      No      00000001
576      services.exe      EtwRegistration 0xe58614efc400    2e35aaeb-857f-4beb-a418-2e6c0e54d988    0      No      00000001
```

By default, about 200 ETW providers are used.
A single process uses a large number of ETW providers.

Tracing ETW from Memory

Tracing ETW Consumer using Volatility

```
ir-mbp:volatility3 tomonaga$  
ir-mbp:volatility3 tomonaga$ python3 vol.py -f test.mem -p plugin etwscan.etwConsumer  
Volatility 3 Framework 2.7.0  
Progress: 100.00 PDB scanning finished
```

PID	ImageFileName	type_map	LoggerId	LoggerName	LogFileName	Guid
700	svchost.exe	EtwConsumer	17	UBPM		c09355a3-96af-4e8f-8d32-a2658dc2d5be
992	svchost.exe	EtwConsumer	10	EventLog-System		d2112be4-cd15-5a9c-e38f-080a207e08d5
992	svchost.exe	EtwConsumer	9	EventLog-Application		c4a0a2bc-c743-5810-8ad4-2655a8ca2744
992	svchost.exe	EtwConsumer	3	Eventlog-Security		0e66e20b-b802-ba6a-9272-31199d0ed295
1332	svchost.exe	EtwConsumer	7	DiagLog		08b524eb-a2bf-47eb-aef1-dbd871741d7a
1964	svchost.exe	EtwConsumer	22	WFP-IPsec Diagnostics	C:\ProgramData\Microsoft\Windows\wfp\wfpdiag.etl	b40325fe-7106-42ac-849e-8aa81df5cb01
1568	svchost.exe	EtwConsumer	8	Diagtrack-Listener		11d8a17b-f2d8-4733-b41b-6f4959acd701
3764	SgrmBroker.exe	EtwConsumer	27	SgrmEtwSession		92ac94a4-8b4f-4055-af12-c30c784da8f0

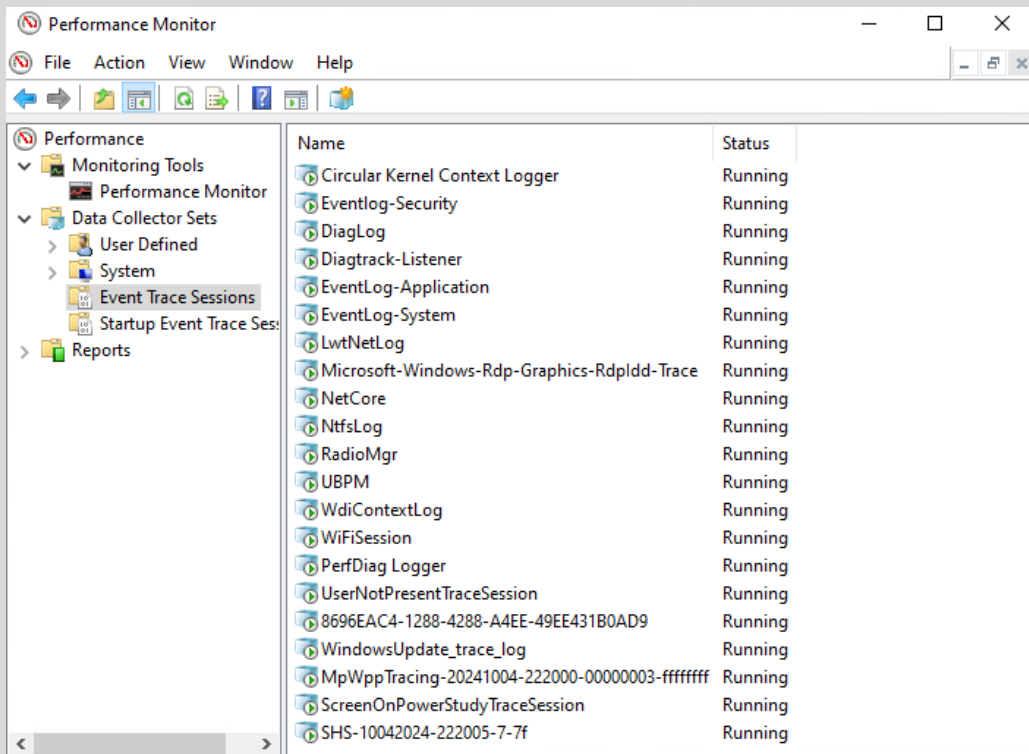
```
ir-mbp:volatility3 tomonaga$
```



All ETW settings(including undocument) can be traced from the memory image.

EDR/AV using ETW

Where is the ETW session for EDR/AV?



EDR/AV using ETW

Where is the ETW session for EDR/AV?

```
Volatility 3 Framework 2.7.1
Progress: 100.00          PDB scanning finished
PID      ImageFileName      TypeMap LoggerId      LoggerName      LogFileName      Guid      Mode
900      svchost.exe        EtwConsumer  17      UBPM            c09355a3-96af-4e8f-8d32-a2658dc2d5be  0x10800190
640      svchost.exe        EtwConsumer  10      EventLog-System d2112be4-cd15-5a9c-e38f-080a207e08d5  0x98800180
640      svchost.exe        EtwConsumer  3       Eventlog-Security 0e66e20b-b802-ba6a-9272-31199d0ed295  0x188001c0
640      svchost.exe        EtwConsumer  9       EventLog-Application c4a0a2bc-c743-5810-8ad4-2655a8ca2744  0x19800180
1900     svchost.exe        EtwConsumer  22     WFP-IPsec Diagnostics C:\ProgramData\Microsoft\Windows\wfp\wfpdiag.etl  b
40325fe-7106-42ac-849e-8aa81df5cb01  0x10802102
1936     svchost.exe        EtwConsumer  7       DiagLog         08b524eb-a2bf-47eb-ae11-dbd871741d7a  0x10800180
1312     svchost.exe        EtwConsumer  8       Diagtrack-Listener 11d8a17b-f2d8-4733-b41b-6f4959acd701  0x8800110
2360     ModuleCoreServ    EtwConsumer  30     McAfee-PCBoost-Monitor 5857d450-9b46-4c17-a2ec-1bf780f73f95  0x800100
8724     SsrnBroker.exe    EtwConsumer  34     SsrnEtwSession  92ac94a4-8b4f-4055-af12-c30c784da8f0  0x800180
14812    MfeAVSvc.exe      EtwConsumer  28     McAfeeRealProtect 420997a8-c91b-4d38-b5cc-37b3da2de1b0  0x2800
```

➔ EDR/AV ETW sessions can only be checked from memory forensics or kernel mode.

AV ETW Session List

Microsoft Defender

- DefenderAuditLogger
- DefenderApiLogger

Kaspersky

- 28E38D72-F060-45EB-B3AE-CAD4834C3A1C
- Eventlog-Security
- {6A7019B4-156F-4092-BC31-828DFA159B56}

ESET

- Eset-Windows-Audit-CVE
- Eset-Threat-Intelligence-Session

McAfee

- McAfee-PCBoost-Monitor
- McAfeeRealProtect
- McAfee-Mmss
- McAfee.{E4367DA7-2B80-47f3-86D2-7626A18FC6F4}
- CSP.{02063ec6-2498-4fff-a32f-d3d693784a18}

EDR ETW Session List

Cylance

- CR_AP_TRACE_SESSION_EX3
- CR_AP_TRACE_SESSION_EX
- [Random Logger Name]
- CR_AP_TRACE_SESSION_EX2
- CR_AP_TRACE_SESSION

Demo

(vol) test@test:~/volatility3\$

Presentation Topics

1

What is ETW?

2

ETW Internals

3

ETW using Incident Response

4

Attack Surface

5

Mitigation and Detection

How to Get New ETW Event

1. Create Event Trace Session

2. Select ETW Provider

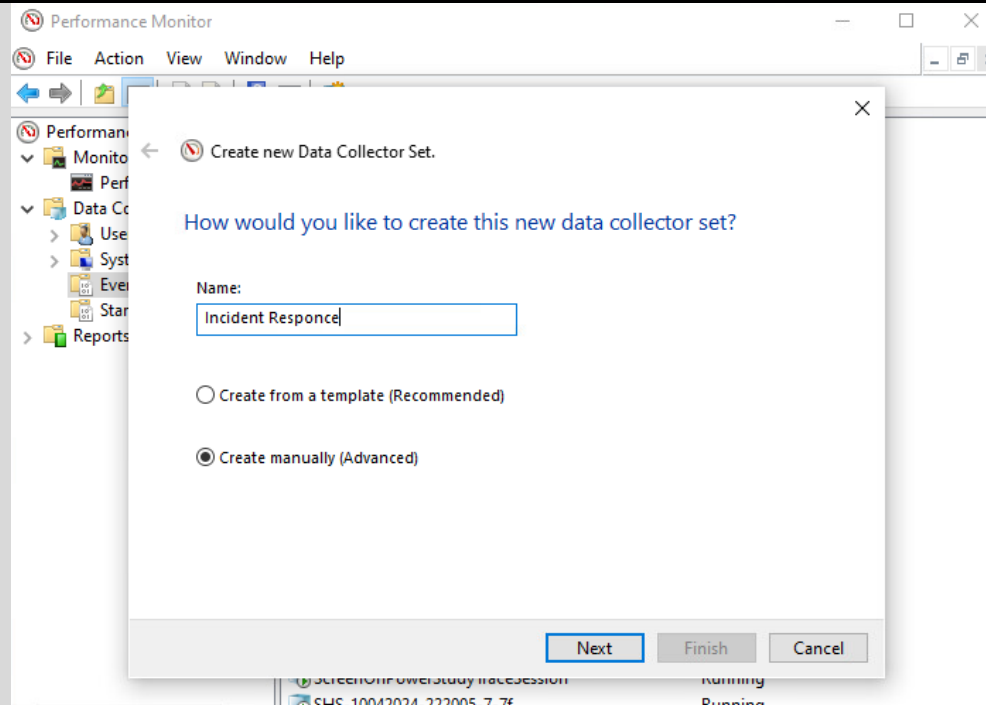
3. Custom Event Trace Session Property

4. Start Event Trace Session

5. Get Event

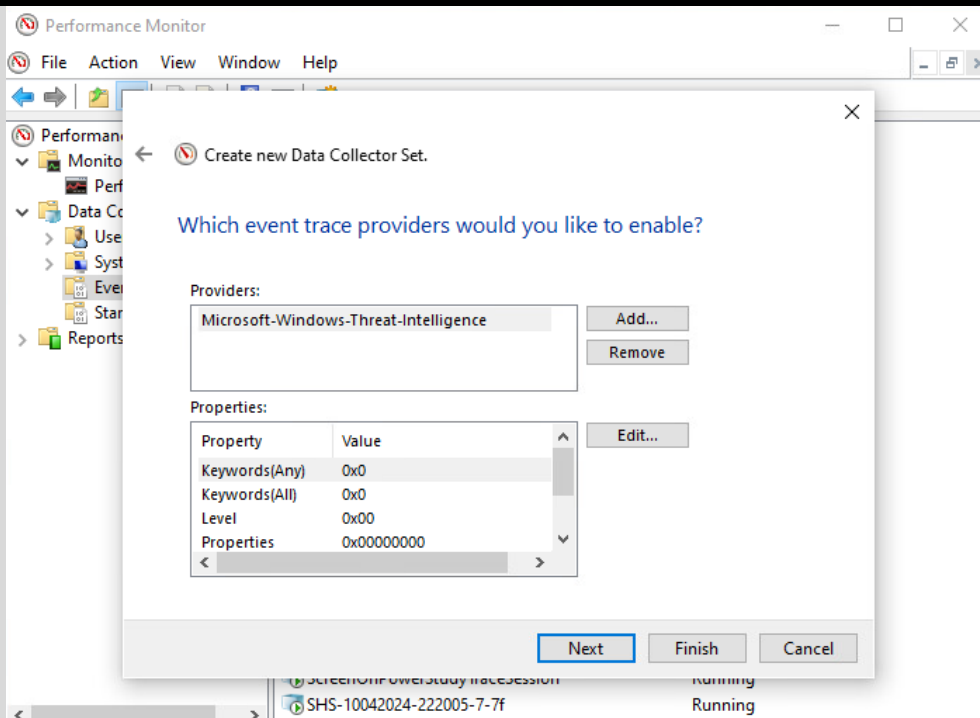
1. Create Event Trace Session

Performance Monitor (perfmon) > Data Collector Set > Event Trace Sessions > New(Data Collector Set)



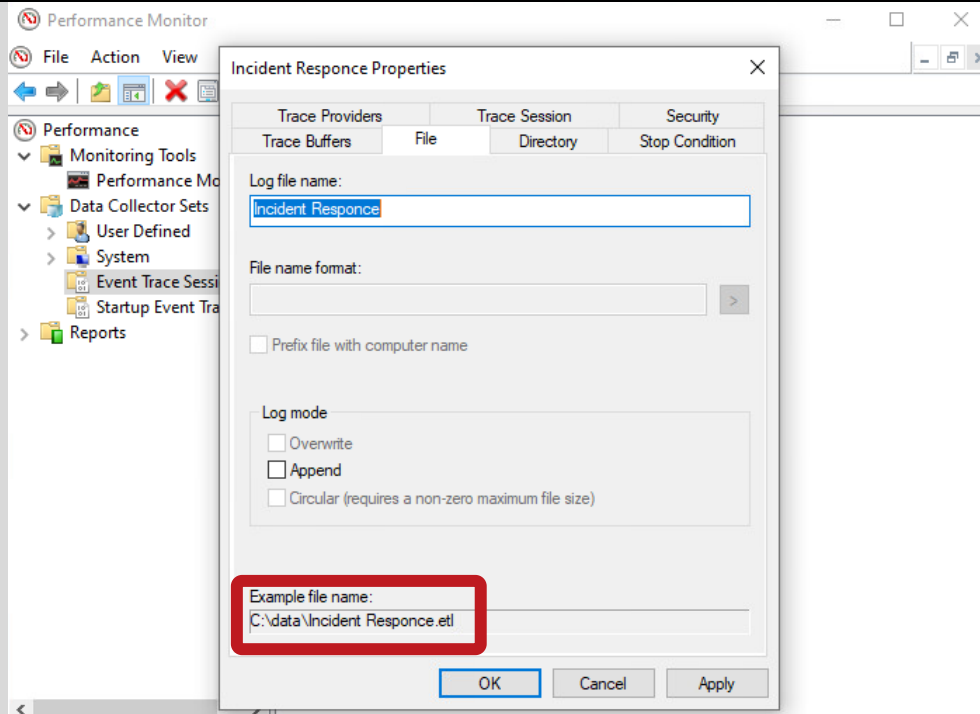
2. Select ETW Provider

Performance Monitor (perfmon) > Data Collector Set > Event Trace Sessions > New(Data Collector Set)



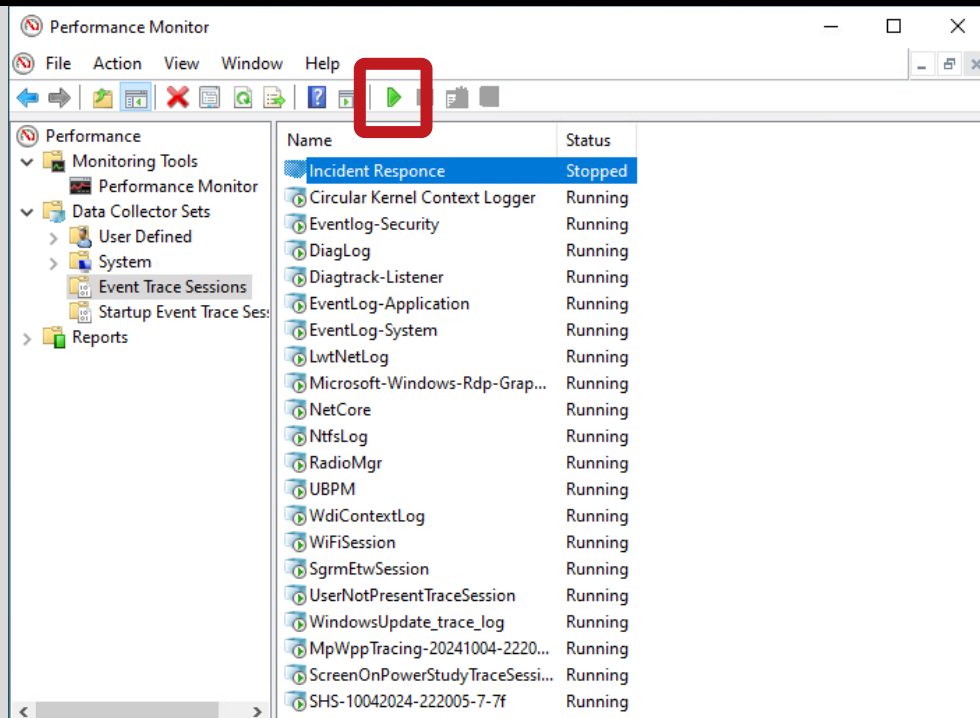
3. Custom Event Trace Session Property

Property > File > Log File Name

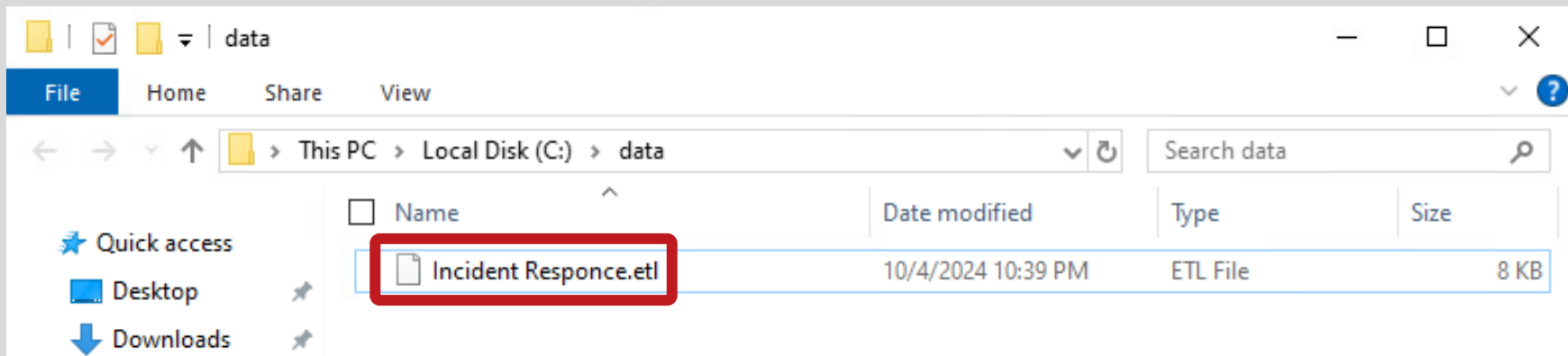


4. Start Event Trace Session

Menu bar > Start



5. Get Event



ETW to EVTX

```
tracert MyFile.etl -o MyFile.etl.evtx -of EVTX -lr
```

ETW to JSON

```
ETW2JSON MyFile.etl MyFile.Kernel.etl --output=MyFile.json
```

<https://github.com/microsoft/ETW2JSON>

How to Analysis ETW using PowerShell

Get-WinEvent can parse ETL files as well as EVTX.

```
# Check Messages (if the message includes.)  
> Get-WinEvent -Path .\test.etl -Oldest | Select-Object Id,Message
```

```
Id Message  
--
```

```
0  
7 Base CPU priority of thread 14308 in process 4148 was changed from 8 t...  
7 Base CPU priority of thread 14308 in process 4148 was changed from 9 t...  
3 Thread 8892 (in Process 4148) started.  
3 Thread 15972 (in Process 4148) started.  
3 Thread 9956 (in Process 4148) started.  
3 Thread 9152 (in Process 4148) started.  
3 Thread 12444 (in Process 4148) started.
```

How to Analysis ETW using PowerShell

Get-WinEvent can parse ETL files as well as EVTX.

```
# Check Messages (if the message includes.)  
> Get-WinEvent -Path .\test.etl -Oldest | Select-Object Id,Message
```

```
Id Message  
--  -
```

```
0  
21  
21  
15  
15  
15  
15  
15
```

How to Analysis ETW using PowerShell

Get-WinEvent can parse ETL files as well as EVTX.

```
# Check Messages (if the message includes.)  
> Get-WinEvent -Path .\test.etl -Oldest | Select-Object Id,Message
```

```
Id Message  
-- -
```

```
0  
21  
21  
15  
15  
15  
15  
15
```

Where is the messages?

How to Analysis ETW using PowerShell

Get-WinEvent can parse ETL files as well as EVTX.

```
> Get-WinEvent -Path .\test.etl -Oldest | Format-List *
Message
Id : 12
Version : 1
Qualifiers
Level : 4
Task : 12
Opcode : 0
Keywords : -9223372036854775648
RecordId : 5
ProviderName : Microsoft-Windows-Kernel-File
ProviderId : edd08927-9cc4-4e65-b970-c2560fb5c289
LogName
ProcessId : 3264
ThreadId : 12728
MachineName : host
UserId
TimeCreated : 2024/05/03 22:20:35
ContainerLog : c:\test.etl
MatchedQueryIds : {}
Bookmark : System.Diagnostics.Eventing.Reader.EventBookmark
LevelDisplayName : 情報
OpcodeDisplayName : 情報
KeywordsDisplayNames : {}
Properties : {System.Diagnostics.Eventing.Reader.EventProperty, System.Diagnostics.Eventing.Reader.EventProperty...}
```

How to Analysis ETW using PowerShell

Get-WinEvent can parse ETL files as well as EVTX.

```
# Search id
> Get-WinEvent -Path .\test.etl -Oldest | Where-Object {$_.ID -eq 12} |
Select-Object -ExpandProperty Properties

```

	Value

	18446623733053668088
	18446623733356940496
	12728
	16908384
	0
...	-assets\hashed-assets\otSearchFeedbackButton-6f435272f9ef2425.js.gz

How to Create Your EDR

EDR uses ETW to detect malicious activity on the Windows OS.



If you understand and use ETW, you can create your own EDR tools (ETW Consumers).

How to Create Your EDR

Steps

1. Find malicious activity to detect

2. Discover ETW Providers

3. Enable ETW Providers

4. Create detection logic using ETW

1. Find malicious activity to detect

Detect activity in NTDS dumps

The screenshot shows the MITRE ATT&CK website interface. The top navigation bar includes links for Matrices, Tactics, Techniques, Defenses, CTI, Resources, and Benefactors. A search bar is present. A notification banner states: "ATT&CK v15.1 has been released! Check out the [blog post](#) or [release notes](#) for more information." Below this, a breadcrumb trail reads: "Home > Techniques > Enterprise > OS Credential Dumping > NTDS". The main heading "OS Credential Dumping: NTDS" is highlighted with a red box. A dropdown menu shows "Other sub-techniques of OS Credential Dumping (8)". The main content area contains a paragraph: "Adversaries may attempt to access or create a copy of the Active Directory domain database in order to steal credential information, as well as obtain other information about domain members such as devices, users, and access rights. By default, the NTDS file (NTDS.dit) is located in %SystemRoot%\NTDS\Ntds.dit of a domain controller.^[1]" followed by another paragraph: "In addition to looking for NTDS files on active Domain Controllers, adversaries may search for backups that contain the same or similar information.^[2]" and a final paragraph: "The following tools and techniques can be used to enumerate the NTDS file and the contents of the entire Active Directory hashes." Below this is a list item: "• Volume Shadow Copy". On the right side, a metadata box contains: "ID: T1003.003", "Sub-technique of: T1003", "Tactic: Credential Access", "Platforms: Windows", "System Requirements: Access to Domain Controller or backup", "Contributors: Ed Williams, Trustwave, SpiderLabs", "Version: 1.2", "Created: 11 February 2020", and "Last Modified: 28 July 2023".

1. Find malicious activity to detect

Detect activity in NTDS dumps

The screenshot shows the MITRE ATT&CK framework interface. The top navigation bar is red and contains the MITRE ATT&CK logo, a search bar, and several menu items: Matrices, Tactics, Techniques, Defenses, CII, Resources, and Benefactors. Below the navigation bar, there is a table with columns for ID, Category, and Description. The table contains one entry for DS0022, which is categorized as 'File' and 'File Access'. The description for DS0022 is: 'Monitor for access or copy of the NTDS.dit.' Below the description, there are two paragraphs of text. The first paragraph is a note: 'Note: Events 4656 and 4663 (Microsoft Windows Security Auditing) provide context of processes and users requesting access or accessing file objects (ObjectType = File) such as C:\Windows\NTDS\ntds.dit. It is important to note that, in order to generate these events, a System Access Control List (SACL) must be defined for the ntds.dit file. Access rights that allow read operations on file objects and its attributes are %%4416 Read file data, %%4419 Read extended file attributes, %%4423 Read file attributes. If you search for just the name of the file and not the entire directory, you may get access events related to the ntds.dit file within a snapshot or volume shadow copy.' The second paragraph is: 'Events 4656 and 4663 (Microsoft Windows Security Auditing) provide context of processes and users creating or copying file objects (ObjectType = File) such as C:\Windows\NTDS\ntds.dit. It is important to note that, in order to generate these events, a System Access Control List (SACL) must be defined for the ntds.dit file. In order to filter file creation events, filter access rights %%4417 Write data to the file and %%4424 Write file attributes.'

ID	Category	Description
DS0022	File	<p>File Access</p> <p>Monitor for access or copy of the NTDS.dit.</p> <p>Note: Events 4656 and 4663 (Microsoft Windows Security Auditing) provide context of processes and users requesting access or accessing file objects (ObjectType = File) such as C:\Windows\NTDS\ntds.dit. It is important to note that, in order to generate these events, a System Access Control List (SACL) must be defined for the ntds.dit file. Access rights that allow read operations on file objects and its attributes are %%4416 Read file data, %%4419 Read extended file attributes, %%4423 Read file attributes. If you search for just the name of the file and not the entire directory, you may get access events related to the ntds.dit file within a snapshot or volume shadow copy.</p> <p>Events 4656 and 4663 (Microsoft Windows Security Auditing) provide context of processes and users creating or copying file objects (ObjectType = File) such as C:\Windows\NTDS\ntds.dit. It is important to note that, in order to generate these events, a System Access Control List (SACL) must be defined for the ntds.dit file. In order to filter file creation events, filter access rights %%4417 Write data to the file and %%4424 Write file attributes.</p>

2. Discover ETW Providers

Find a provider to trace CreateFile

```
> logman query providers| Select-String "file"

...
Microsoft-Windows-FileHistory-Core      {B447B4DB-7780-11E0-ADA3-18A90531A85A}
Microsoft-Windows-FileHistory-Engine    {B447B4DE-7780-11E0-ADA3-18A90531A85A}
Microsoft-Windows-FileHistory-EventListener {B447B4DF-7780-11E0-ADA3-
18A90531A85A}
Microsoft-Windows-FileHistory-Service   {B447B4E0-7780-11E0-ADA3-18A90531A85A}
Microsoft-Windows-FileHistory-UI        {B447B4E1-7780-11E0-ADA3-18A90531A85A}
Microsoft-Windows-FileInfoMinifilter    {A319D300-015C-48BE-ACDB-47746E154751}
Microsoft-Windows-Kernel-File         {EDD08927-9CC4-4E65-B970-C2560FB5C289}
Microsoft-Windows-NetworkProfile        {FBCFAC3F-8459-419F-8E48-1F0B49CDB85E}
...
```

3. Enable ETW Providers

Enable ETW Providers using PowerShell

```
> New-NetEventSession -Name "FileActivity" -LocalFilePath "%TEMP%\FileActivity.etl"  
  
> Add-NetEventProvider -Name " Microsoft-Windows-Kernel-File" -SessionName  
"FileActivity"  
  
> Start-NetEventSession -Name "FileActivity"  
  
> Stop-NetEventSession -Name "FileActivity"
```

4. Create detection logic using ETW

Detect suspicious file activity with PowerShell

```
$SearchFileName = "%SystemRoot%\NTDS\Ntds.dit"
$LogName = ".\FileTrace.etl"
$Interval = 60

while ($true) {
    $events = Get-WinEvent -Path $LogName -Oldest | Where-Object {$_.ID -eq 12} |
        Select-Object -ExpandProperty Properties | Where-Object {$_.Value -like $SearchFileName} |
        Select-Object -ExpandProperty Value

    if ($events) {
        foreach ($event in $events) {
            $message = "Access to $event"
            Write-Host $message -ForegroundColor Red
        }
    } else {
        Write-Host "No events" -ForegroundColor Green
    }

    Start-Sleep -Seconds $Interval
}
```

Tips: How to programming the detection logic?

Answer

Please ask ChatGPT!

Please create a sample program in PowerShell that detects that the NTDS.dit file has been recorded in the properties column of the sample.etl file that contains the ETW event.



To detect that the `NTDS.dit` file is mentioned in the **Properties** column of events in an **ETL (Event Trace Log)** file using PowerShell, you can write a script that reads the ETL file, checks the properties of each event, and looks for references to the `NTDS.dit` file.

The `NTDS.dit` file is a critical database for Active Directory Domain Services (AD DS), so events related to it could indicate sensitive actions like backup, access, or replication. In this example, the script will search for the string `NTDS.dit` in the **Properties** field of the events.

Here's a sample PowerShell script that achieves this:

Sample PowerShell Script

```
powershell コードをコピーする

# Define the path to the ETL file
$etlFilePath = "C:\Path\To\sample.etl"

# Define the search string (NTDS.dit)
$searchString = "NTDS.dit"

# Load the events from the ETL file
try {
    Write-Host "Loading events from ETL file: $etlFilePath" -ForegroundColor Green
    $events = Get-WinEvent -Path $etlFilePath
```

How the Script Works:

- Specify the ETL file path:** The script begins by specifying the path of the ETL file (`$etlFilePath`) to be processed.
- Define the search string:** The script looks for any reference to `NTDS.dit` in the **Properties** of each event. The search string is set to `"NTDS.dit"`.
- Load the ETL file:** Using `Get-WinEvent -Path $etlFilePath`, the script loads the ETL file into memory.
- Search within Event Properties:** The script filters the events by checking if any of the event's **Properties** contains the string `"NTDS.dit"`. This is done using the `Where-Object` cmdlet to examine the properties.
- Output matching events:** If any events contain `NTDS.dit` in their properties, the matching events will be displayed with their **TimeCreated**, **ID**, **ProviderName**, **Message**, and **Properties** fields.
- Optional CSV Export:** The script exports the filtered events to a CSV file (`FilteredEvents.csv`) for further analysis if needed.

How to Use the Script:

- Update the paths:**
 - Set the correct ETL file path by updating the `$etlFilePath` variable.
 - Optionally, update the `$csvOutputPath` variable to change the output path for the CSV file.
- Run the script:**

Sysmon vs ETW

	Sysmon	ETW
Advantage	✓ Easy to setup	✓ Installed by default ✓ More data than Sysmon
Disadvantage	✓ Not installed by default	✓ limited information on how to use
Use case	✓ Log archive and analysis	✓ Real-time detection

If you want to create a real-time detection system like EDR, **ETW** is recommended.



If you want an incident response tool, **Sysmon** is recommended.

ETW in Forensics

Question

(If the etl file has been deleted) Can ETW events be **recovered**?

ETW Carving from Memory

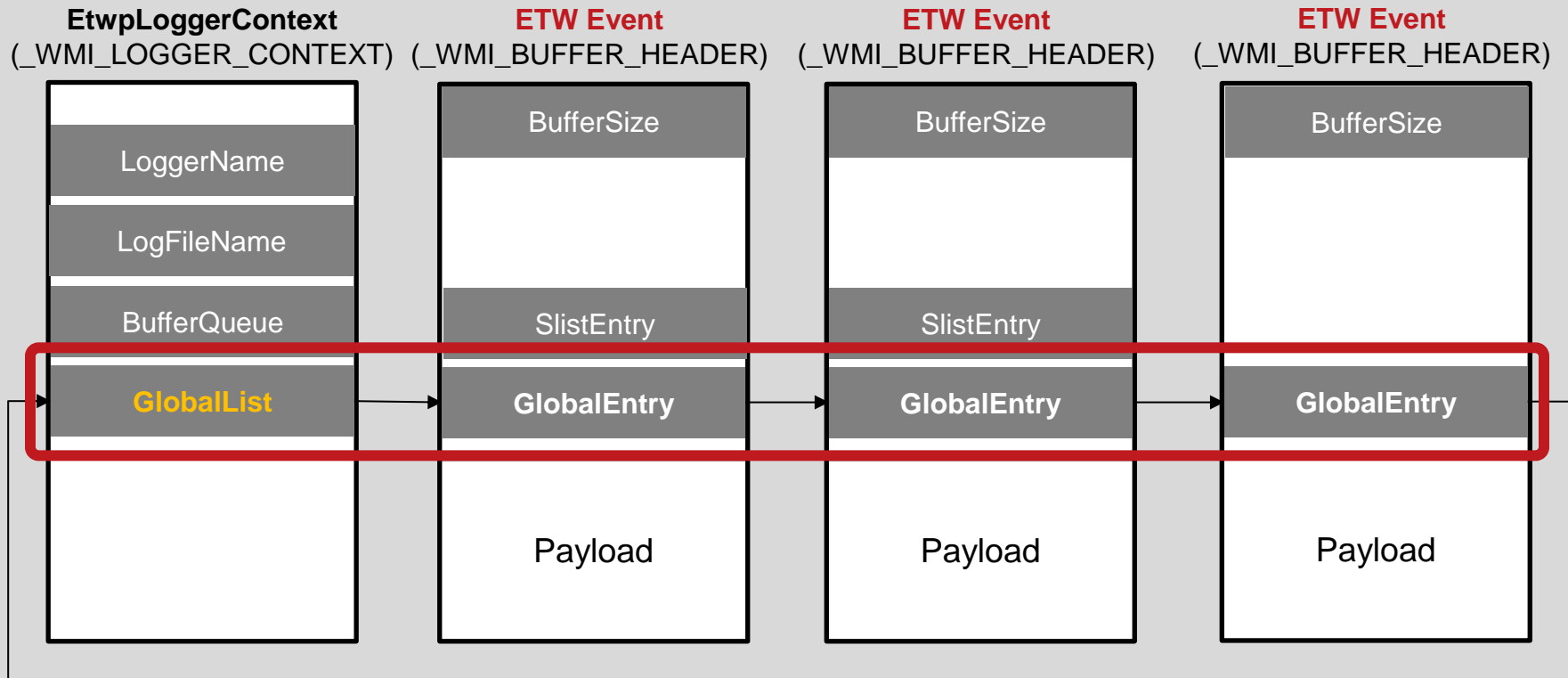
ETL file format

Hex Address	Hex Data	ASCII Data
00000000	00 20 00 00 60 02 00 00 60 02 00 00 00 00 00 00
00000001	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000002	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000003	60 02 00 00 21 00 04 00 00 00 00 00 00 00 00 00	...!
00000040	00 00 00 00 00 00 00 00 02 00 02 C0 C2 01 00 00 00AA...
00000050	44 3B 00 00 18 29 00 00 95 CF 9F 3D 71 0B 00 00 00	D;...)*ÿ=q...
00000060	5C 08 00 00 4D 04 00 00 00 20 00 00 0A 00 01 05	\...M.....
00000070	65 4A 00 00 08 00 00 00 76 4E A7 59 78 8A DA 01	eJ.....vNSYxŠÚ.
00000080	5A 62 02 00 00 00 00 00 05 00 00 00 01 00 00 00	Zb.....
00000090	01 00 00 00 08 00 00 00 00 00 00 00 00 09 00 00
000000A0	09 00 00 00 00 00 00 00 06 00 00 00 00 00 00 00
000000B0	E4 FD FF FF 40 00 74 00 7A 00 72 00 65 00 73 00	äyyy@.t.z.r.e.s.
000000C0	2E 00 64 00 6C 00 6C 00 2C 00 2D 00 36 00 33 00	..d.l.l.,.-.6.3.
000000D0	32 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	2.....
000000E0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000F0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

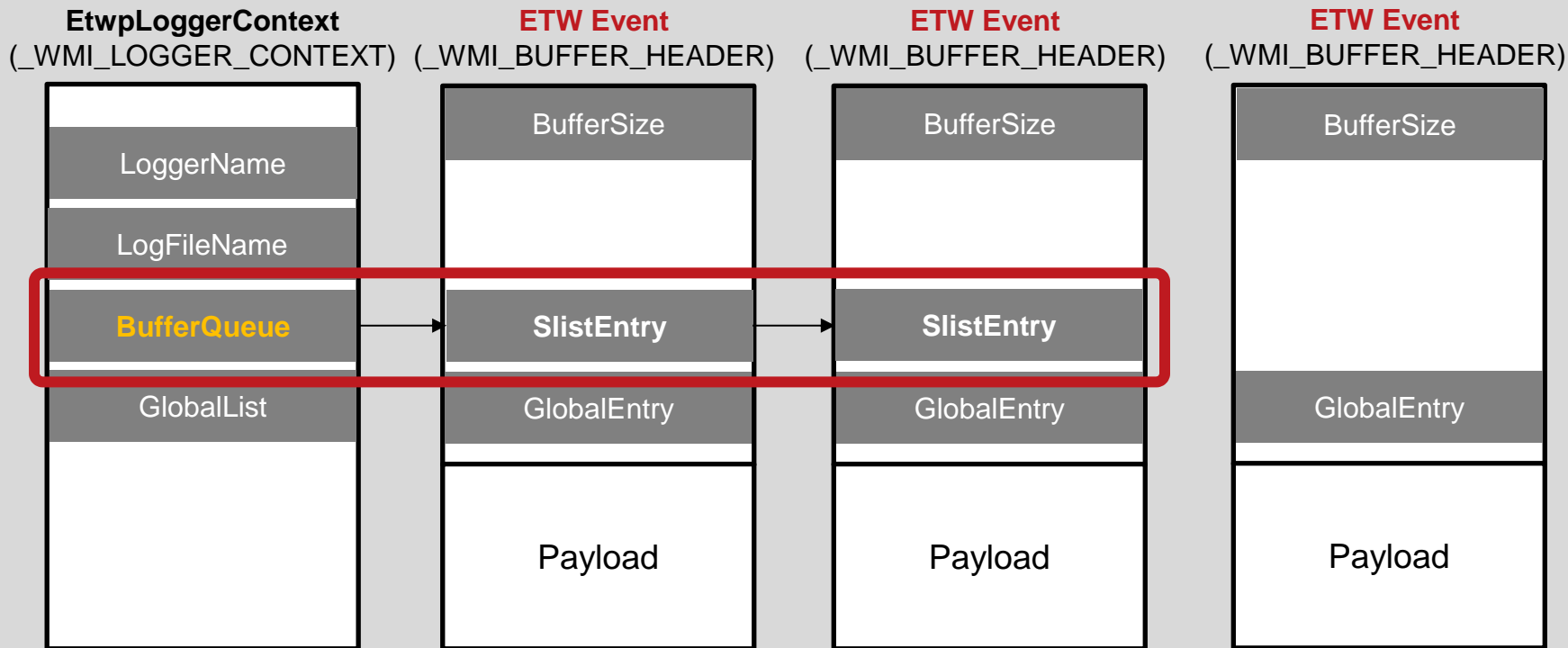
No signature = Carving cannot be used

00000160	40 9F DE 90 00 7A 01 80 96 98 00 00 00 00 00 00	0Yb".ú.é-.....
00000170	63 0D 2F C9 77 8A DA 01 01 00 00 00 00 00 00 00	c./éwŠÚ.....
00000180	49 00 6E 00 63 00 69 00 64 00 65 00 6E 00 74 00	I.n.c.i.d.e.n.t.
00000190	20 00 52 00 65 00 73 00 70 00 6F 00 6E 00 73 00	.R.e.s.p.o.n.s.
000001A0	65 00 00 00 43 00 3A 00 5C 00 55 00 73 00 65 00	e...C.:.\U.s.e.
000001B0	72 00 73 00 5C 00 6B 00 61 00 6E 00 72 00 69 00	r.s.\.k.a.n.r.i.
000001C0	5C 00 41 00 70 00 70 00 44 00 61 00 74 00 61 00	\.A.p.p.D.a.t.a.
000001D0	5C 00 4C 00 6F 00 63 00 61 00 6C 00 5C 00 49 00	\.L.o.c.a.l.\.I.
000001E0	6E 00 63 00 69 00 64 00 65 00 6E 00 74 00 20 00	n.c.i.d.e.n.t. .
000001F0	52 00 65 00 73 00 70 00 6F 00 6E 00 73 00 65 00	R.e.s.p.o.n.s.e.
00000200	2E 00 65 00 74 00 6C 00 00 00 00 00 00 00 00 00	..e.t.l.....
00000210	02 00 02 C0 50 00 50 00 44 3B 00 00 18 29 00 00	...ÄP.P.D;...)
00000220	95 CF 9F 3D 71 0B 00 00 5C 08 00 00 4D 04 00 00	*ÿ=q...\...M...
00000230	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000240	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000250	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000260	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
00000270	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy

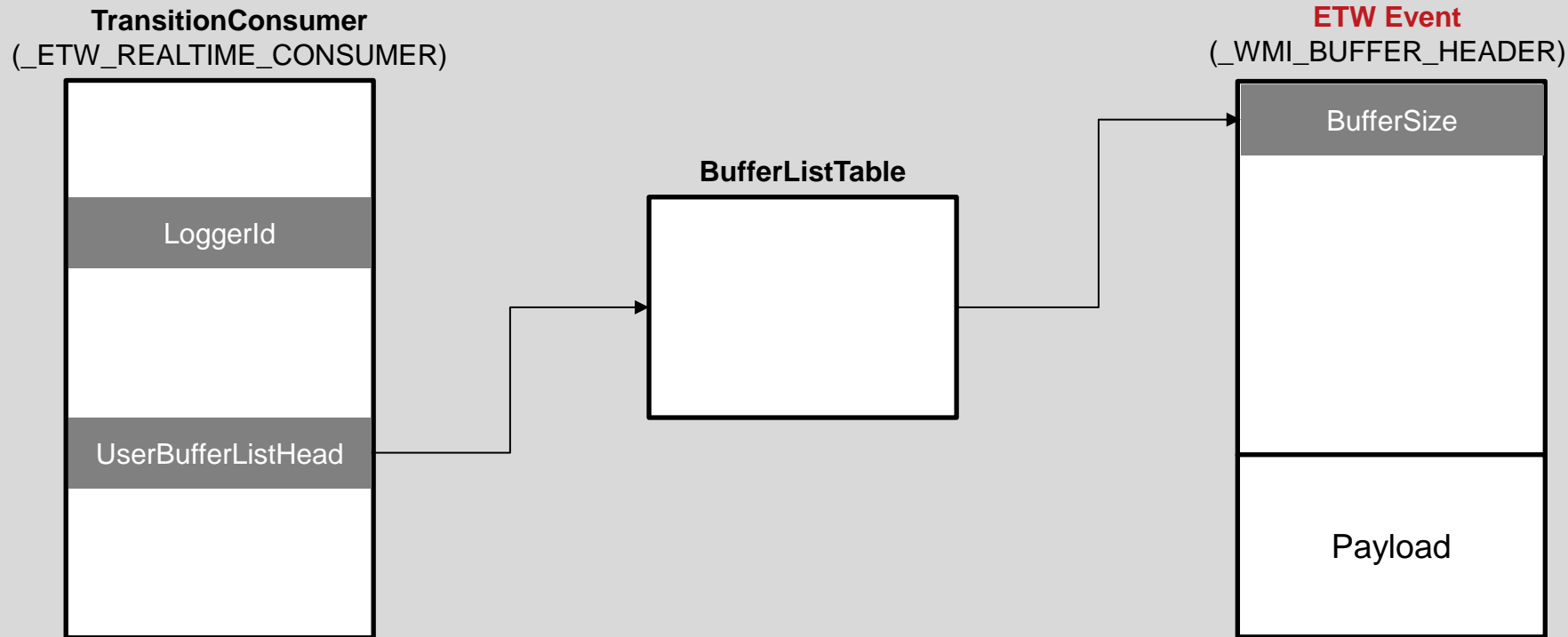
Trace ETW Data from the Structure



Trace ETW Data from the Structure



Trace ETW Data from the Structure



What is the difference between GlobalList and BufferQueue?

GlobalList _WMI_LOGGER_CONTEXT

- All cached ETW events.

BufferQueue _WMI_LOGGER_CONTEXT

- Queue of ETW events to be written to the ETL file.

- **Stream mode: File**

- If ETW events are not saved in the ETL file, the BufferQueue is not used.

UserBufferListHead _ETW_REALTIME_CONSUMER

- Tracing ETW events to Consumers in real time.

- **Stream mode: Real time**

What is the difference between GlobalList and BufferQueue?

GlobalList _WMI_LOGGER_CONTEXT

- All cached ETW events.

BufferQueue _WMI_LOGGER_CONTEXT

- Queue of ETW events to be written to the ETL file.

- **Stream mode: File**

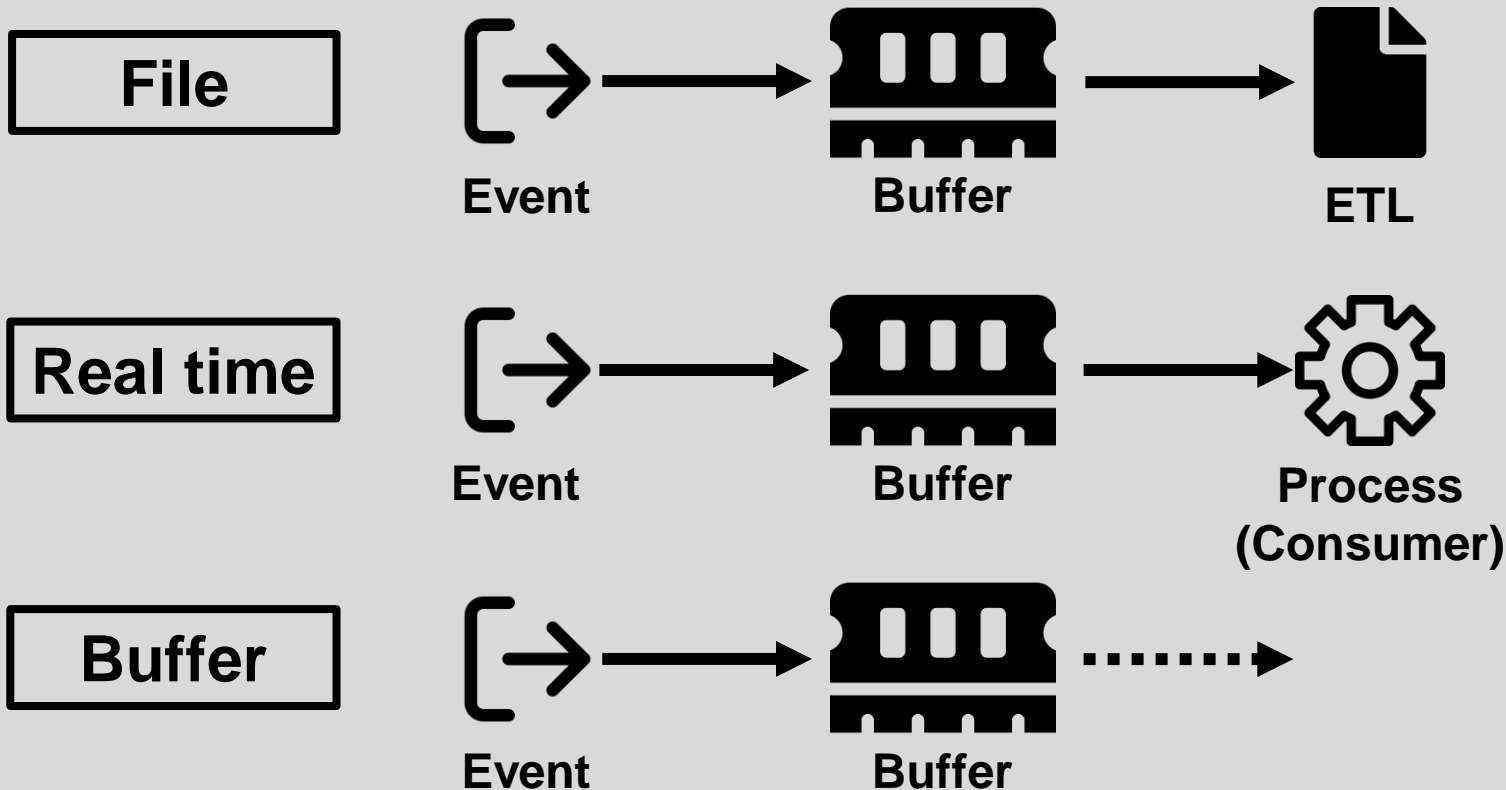
- If ETW events are not saved in the ETL file, the BufferQueue is not used.

UserBufferListHead _ETW_REALTIME_CONSUMER

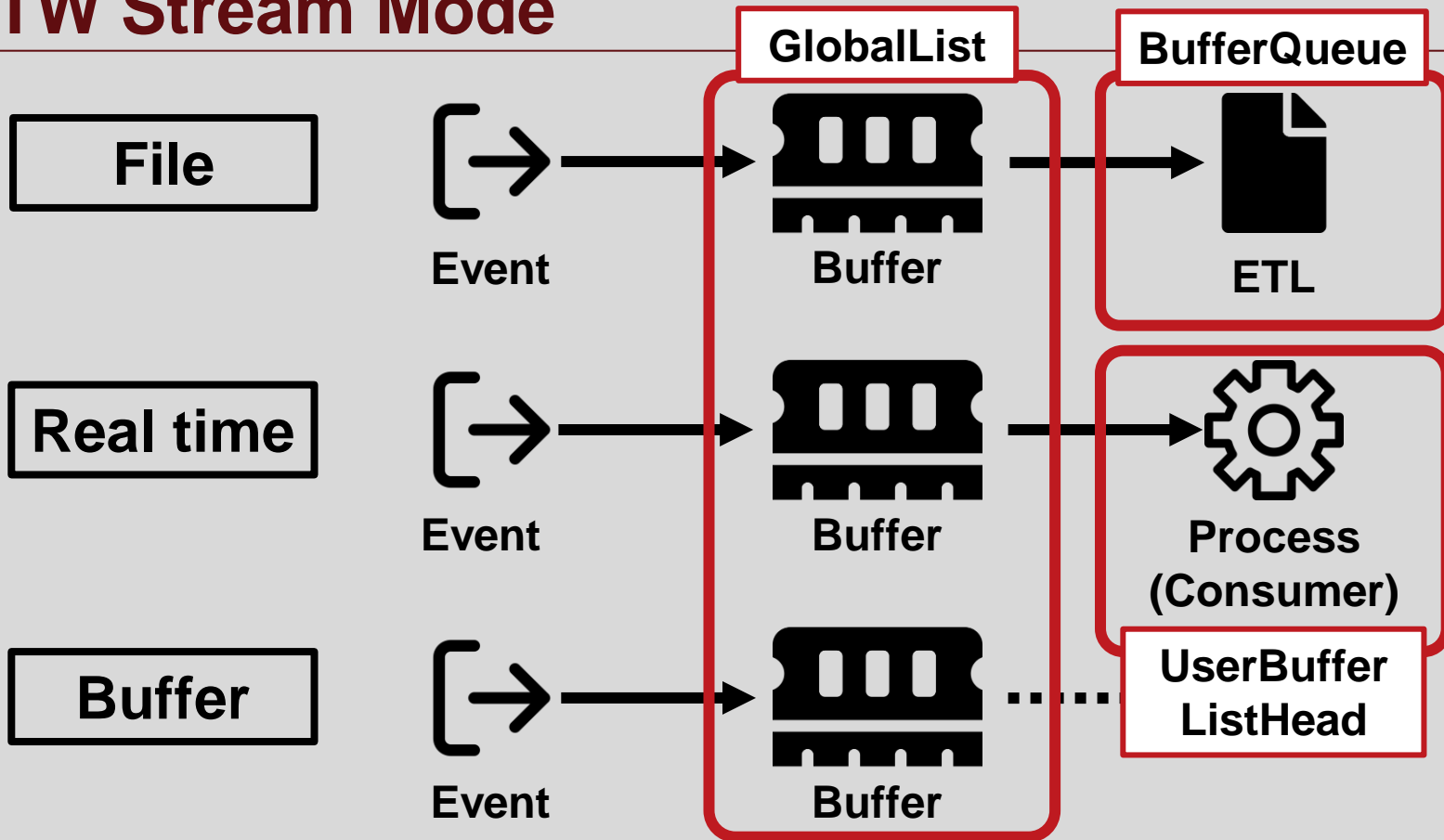
- Tracing ETW events to Consumers in real time.

- **Stream mode: Real time**

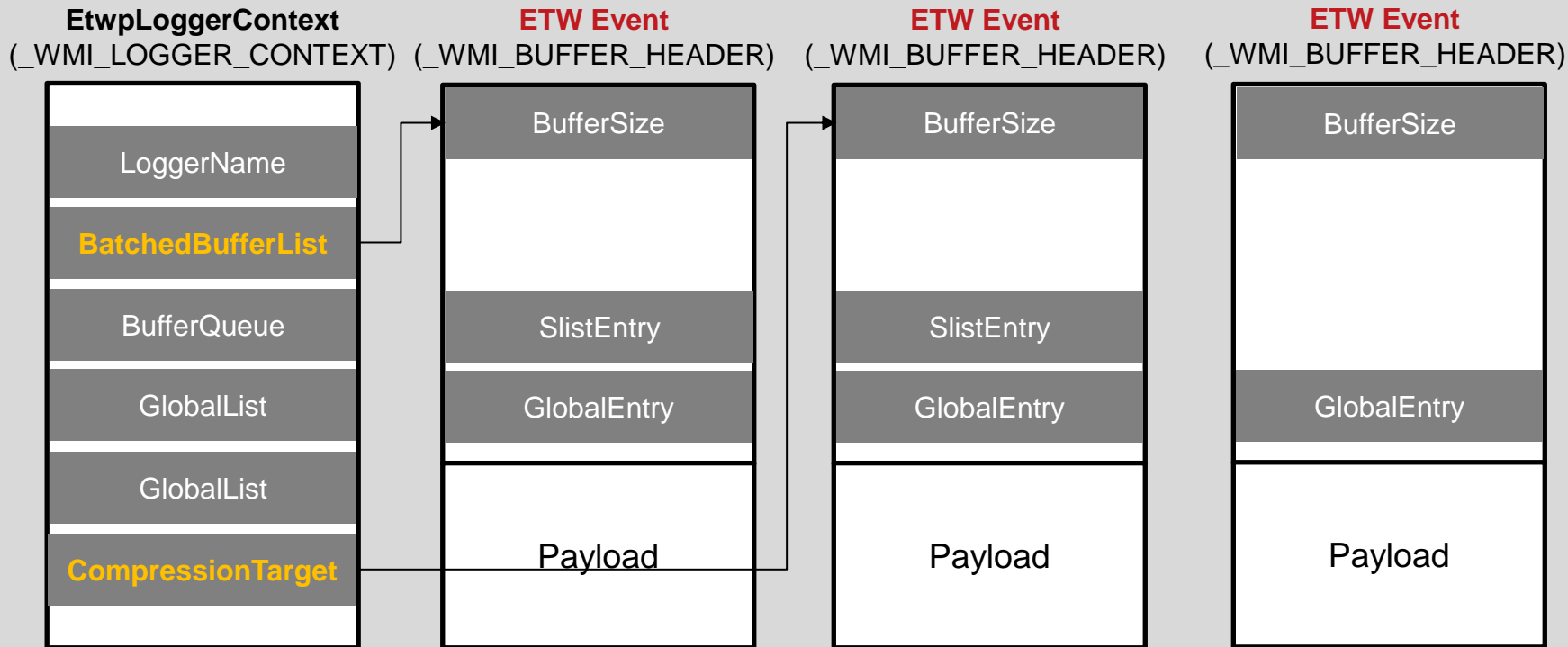
ETW Stream Mode



ETW Stream Mode



Trace ETW Data from the Structure



Dump ETW from Memory

Volatility3 plugin

```
Volatility 3 Framework 2.7.0
Progress: 100.00 PDB scanning finished
PID ImageFileName type_map LoggerId LoggerName LogFileName Guid Mode
848 svchost.exe EtwConsumer 17 UBPM c09355a3-96af-4e8f-8d32-a2658dc2d5be 0x10800190
1836 svchost.exe EtwConsumer 13 EventLog-System d2112be4-cd15-5a9c-e38f-080a207e08d5 0x10800180
1836 svchost.exe EtwConsumer 10 EventLog-Application c4a0a2bc-c743-5810-8ad4-2655a8ca2744 0x11800180
1836 svchost.exe EtwConsumer 3 EventLog-Security 0e66e20b-b802-ba6a-9272-31199d0ed295 0x108001c0
1844 svchost.exe EtwConsumer 9 DiagLog 08b524eb-a2bf-47eb-ae11-dbd871741d7a 0x10800180
1844 svchost.exe EtwConsumer 21 WFP-IPsec Diagnostics C:\ProgramData\Microsoft\Windows\wfp\wfpdiag.etl b40325fe-7106-42ac-849e-8aa81df5cb01 0x10802102
1880 svchost.exe EtwConsumer 24 Diagtrack-Listener bd6a694f-11ae-11ee-8e91-000c2962ae37 0x8800110
4 System - 2 Circular Kernel Context Logger 54dea73a-ed1f-42a4-af71-3e63d056f174 0x2800480
4 System - 4 AppModel a922a8be-2450-438e-9520-fbcdfb46b0bd 0x10808400
4 System - 5 Audio 15bc788a-6a38-4d79-8773-b53fdfb84d79 0x10808400
4 System - 6 FileActivity_realtime 75f3a0a4-ced8-4e82-9718-3f4b7b249fa1 0x400100
4 System - 7 DefenderApiLogger 6b4012d0-22b6-464d-a553-20e9618403a2 0x18800180
4 System - 8 DefenderAuditLogger 6b4012d0-22b6-464d-a553-20e9618403a1 0x188001c0
4 System - 11 EventLog-DebugChannel %SystemRoot%\System32\Winevt\Logs\DebugChannel.etl 2533f63b-45f3-5b31-e7c5-82fcf6979473 0x800081
4 System - 12 EventLog-Microsoft-RMS-MSIPC-Debug %SystemRoot%\System32\Winevt\Logs\Microsoft-RMS-MSIPC4Debug.etl 199b952f-e81e-5242-23d4-ae228d121d95 0x8000
82
4 System - 14 LwtNetLog C:\WINDOWS\System32\LogFiles\WMI\LwtNetLog.etl 603ba31e-ec5a-4cde-be87-ed0a16c3b170 0x800002
4 System - 15 NtfsLog 8184e181-19c8-45ab-89d1-d8eaf117208f 0x10808400
4 System - 16 FileActivity_save C:\Users\kanri\AppData\Local\FileActivity_save.etl 93da76d9-d449-40f4-87ca-25963e9bf530 0x400000
4 System - 18 WdiContextLog C:\WINDOWS\System32\Wdi\LogFiles\WdiContextLog.etl.001 f52ac1cc-b92d-4d8e-8cf5-699ca40a73d2 0x800082
4 System - 19 WiFiSession C:\WINDOWS\System32\LogFiles\WMI\Wifi.etl 76e684e4-194c-43b0-b890-8269646de989 0x800002
4 System - 20 UserNotPresentTraceSession C:\WINDOWS\system32\SleepStudy\UserNotPresentSession.etl bd6a6933-11ae-11ee-8e91-806e6f6e6963 0x10800002
4 System - 22 WindowsUpdate_trace_log C:\WINDOWS\Logs\WindowsUpdate\WindowsUpdate.20230623.191504.669.25.etl e11f5e99-f330-4d41-8d1f-150fd6bb2f23 0x11802009
```

```
ir-mbp:volatility3 tomonaga$ ls *.etl
FileActivity_save.0.etl FileActivity_save.16.etl FileActivity_save.4.etl LwtNetLog.2.etl WdiContextLog.2.etl
FileActivity_save.1.etl FileActivity_save.17.etl FileActivity_save.5.etl UserNotPresentTraceSession.0.etl WiFiSession.0.etl
FileActivity_save.10.etl FileActivity_save.18.etl FileActivity_save.6.etl UserNotPresentTraceSession.1.etl WiFiSession.1.etl
FileActivity_save.11.etl FileActivity_save.19.etl FileActivity_save.7.etl UserNotPresentTraceSession.2.etl WiFiSession.2.etl
FileActivity_save.12.etl FileActivity_save.2.etl FileActivity_save.8.etl WFP-IPsec_Diagnostics.0.etl WindowsUpdate_trace_log.0.etl
FileActivity_save.13.etl FileActivity_save.20.etl FileActivity_save.9.etl WFP-IPsec_Diagnostics.1.etl WindowsUpdate_trace_log.1.etl
FileActivity_save.14.etl FileActivity_save.21.etl LwtNetLog.0.etl WdiContextLog.0.etl WindowsUpdate_trace_log.2.etl
FileActivity_save.15.etl FileActivity_save.3.etl LwtNetLog.1.etl WdiContextLog.1.etl
```

Demo

(vol) test@test:~/volatility3\$

ETW Scanner for Volatility3

The screenshot shows the GitHub interface for the repository 'etw-scan' by JPCERTCC. The repository is private and has 6 unwatchers, 0 forks, and 0 stars. The main branch is 'main' with 1 branch and 0 tags. The repository was first committed by 'shu-tom' 26 minutes ago, with 2 commits. The file list includes: `.github/workflows`, `patch`, `plugins`, `volatility3/volatility3/framework/symbols/wind...`, `.gitignore`, `LICENSE.txt`, and `README.md`, all committed 26 minutes ago. The 'About' section describes it as an 'ETW forensic tool for Volatility3 plugin' with tags for security, memory, incident-response, forensics, volatility-plugins, and volatility-framework. The 'Languages' section shows Python at 100.0%. The 'Description' section contains the URL <https://github.com/JPCERTCC/etw-scan> and a partial description: 'This tool is a Volatility3 plugin that scans memory dumps for events matching the incidents query. This tool can extract detailed ETW configuration settings that cannot be checked in user mode. This plugin can recover ETW events (ETL

1

What is ETW?

2

ETW Internals

3

ETW using Incident Response

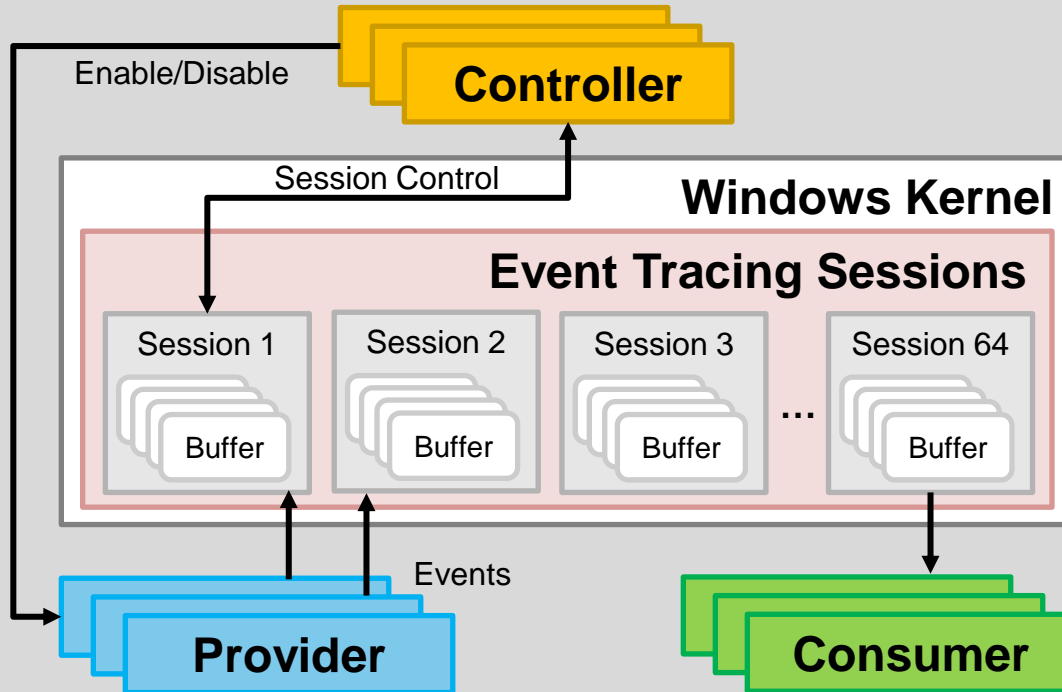
4

Attack Surface

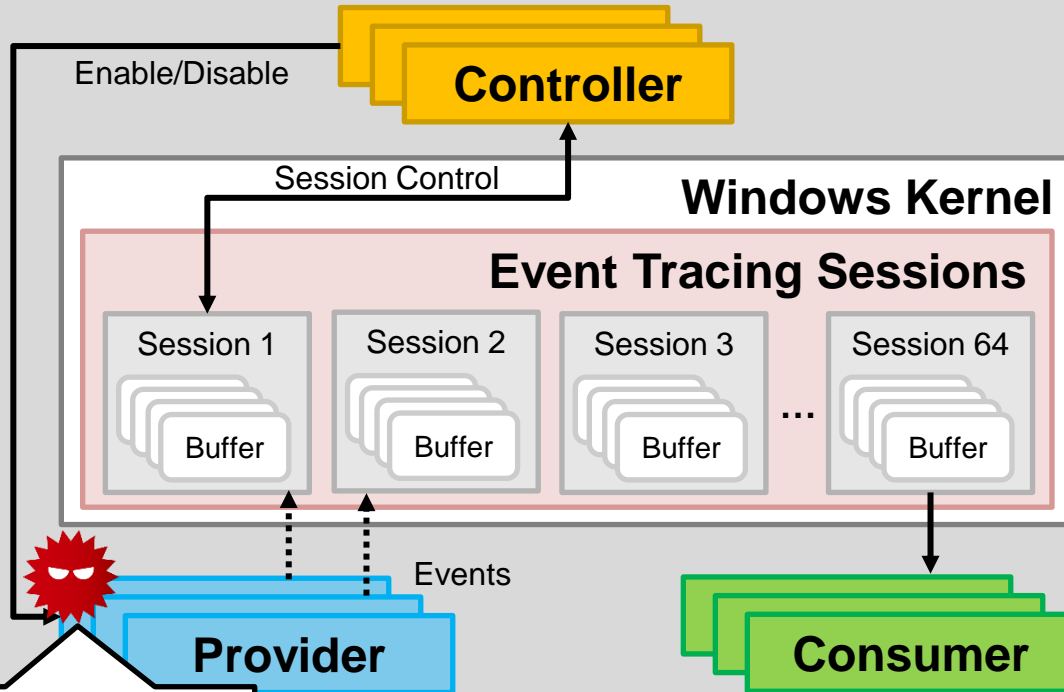
5

Mitigation and Detection

Attack Surface for ETW

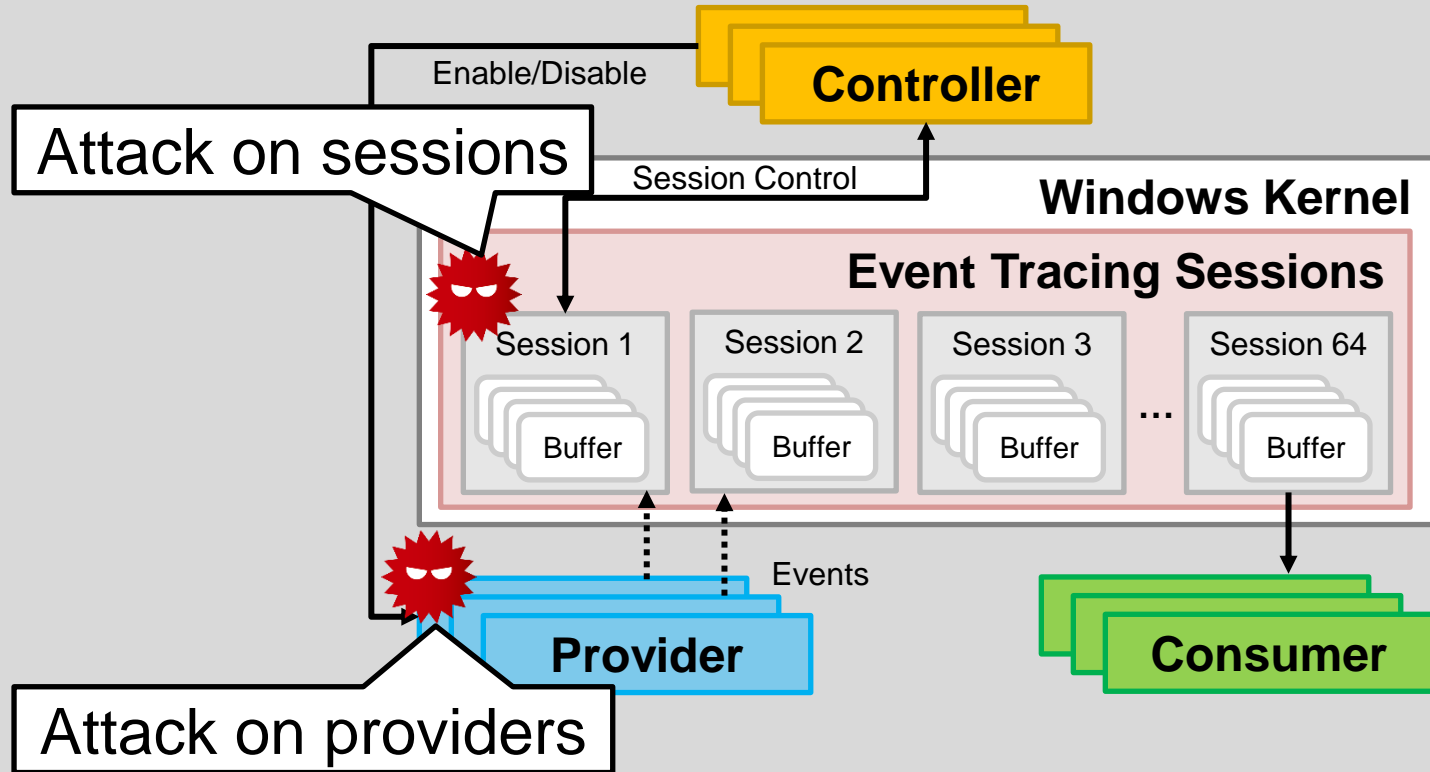


Attack Surface for ETW



Attack on providers

Attack Surface for ETW



ETW Bypass Technique

Attack on providers

- Block or delete the writing of events by ETW providers.

Attack on sessions

- Stop or delete sessions.

Block Event Writing of Providers using API Hook

Intercept the ETW API and disallow the API to execute.

EventWrite

EtwEventWrite

NtTraceEvent

EtwEventWriteFull

ETW Bypass using Malware

HUI Loader using attack providers

```

int mal_ETW_bypass()
{
    HMODULE ModuleHandleA; // rax
    FARPROC EtwEventWrite; // rbx
    HANDLE CurrentProcess; // rax
    HANDLE v3; // rax
    HANDLE v4; // rax
    char Buffer[4]; // [rsp+30h] [rbp-28h] BYREF
    DWORD f1OldProtect; // [rsp+34h] [rbp-24h] BYREF
    CHAR ModuleName[16]; // [rsp+38h] [rbp-20h] BYREF

    Buffer[0] = 0xC3;
    strcpy(ModuleName, "n_d");
    ModuleHandleA = GetModuleHandleA(ModuleName);
    if (ModuleHandleA)
    {
        EtwEventWrite = GetProcAddress(ModuleHandleA, "EtwEventWrite");
        CurrentProcess = GetCurrentProcess();
        VirtualProtectEx(CurrentProcess, EtwEventWrite, 1ui64, 0x40u, &f1OldProtect);
        v3 = GetCurrentProcess();
        WriteProcessMemory(v3, EtwEventWrite, Buffer, 1ui64, 0i64);
        v4 = GetCurrentProcess();
        LODWORD(ModuleHandleA) = VirtualProtectEx(v4, EtwEventWrite, 1ui64, f1OldProtect, 0i64);
    }
    return (int)ModuleHandleA;
}

```



Delete Providers

Using Windows API

- EtwEventUnregister

Overwriting ETW structures

- WMI_LOGGER_CONTEXT
- ETW_GUID_ENTRY
- ETW_REG_ENTRY

Stop or Delete Sessions

Execute ETW API to stop the session.

StopTrace

ControlTrace

NtTraceControl

1

What is ETW?

2

ETW Internals

3

ETW using Incident Response

4

Attack Surface

5

Mitigation and Detection

Mitigation and Detection

API hook detection

- ❑ Check whether ETW API page protections on memory blocks is not **PAGE_EXECUTE_READWRITE**.
- ❑ Check whether the top area of the ETW API has not been modified.

- ❑ Target ETW API:
 - ✓ EventWrite
 - ✓ EtwEventWrite
 - ✓ NtTraceEvent
 - ✓ EtwEventWriteFull

Mitigation and Detection

Disable ETW detection

- ❑ Block escalation of administrator privileges (like vulnerability exploits).
- ❑ Monitor ETW registry and structure changes.
 - ✓ HKLM\Software\Microsoft\.NETFramework\ETWEnabled
 - ✓ _WMI_LOGGER_CONTEXT

Takeaways


The internal structure of ETW is difficult to parse, and so it is better to analyze it with PowerShell or convert it to EVTX.

ETW can be used to create custom EDR.

To recover ETW from memory, structure analysis is necessary, and the Volatility3 Plugin that we have created can be helpful.

ETW bypass will be used in many attacks in future, and so mitigation and detection is needed.

Thank you!

 @jpcert_en
@jpcert_ac

 ir-info@jpcert.or.jp
PGP <https://www.jpcert.or.jp/english/pgp/>