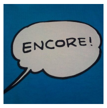
ETW Forensics - Why use Event Tracing for Windows over EventLog? -

blogs.jpcert.or.jp/en/2024/11/etw_forensics.html



朝長 秀誠 (Shusei Tomonaga)

November 14, 2024

volatility

Email

Many people may think of EventLogs when one mentions Windows OS logs. When investigating incidents such as malware infections, it is common to analyze the Windows OS EventLogs to find traces that may help uncover the incident. However, since the EventLog is not designed to detect suspicious behavior on Windows OS, you may not always find the information you are looking for when investigating an incident. Therefore, it is necessary to enable audit logs or install Sysmon to obtain more information.
There is another mechanism in Windows OS that can detect suspicious behavior. It is a feature called Event Tracing for Windows (ETW). This is a system for managing events generated by the kernel and processes, and it is used for debugging applications and other purposes. ETW is also used for collecting and managing EventLogs, and in recent years it has been used in the detection logic of EDR products and antivirus software. ETW has a function that can log various behaviors in the OS as events by default, which makes it possible to obtain more information than EventLogs.
This article explains the structure of ETW and how you can use it for your forensics.

ETW Internals

ETW architecture

Figure 1 shows the components of ETW[1]. Providers such as applications send events, and after they are stored in buffers, consumers such as EDR receive them.

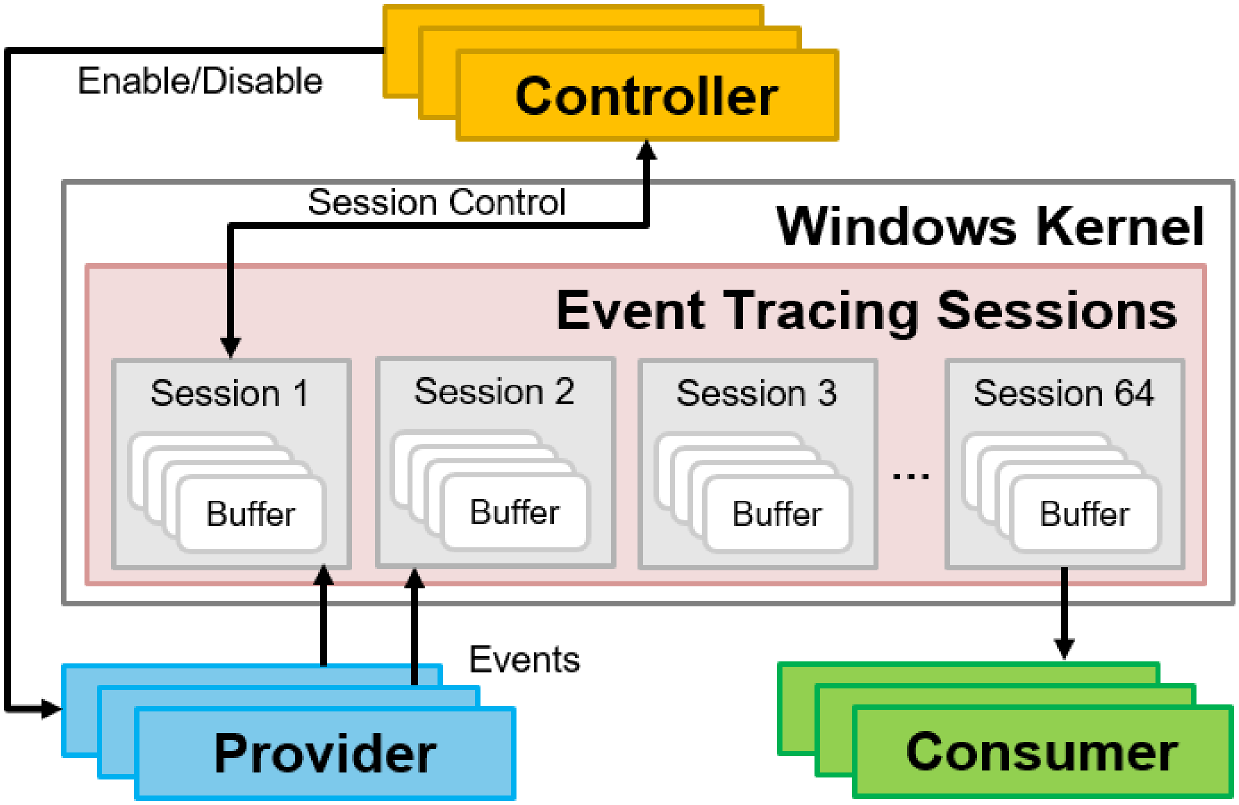
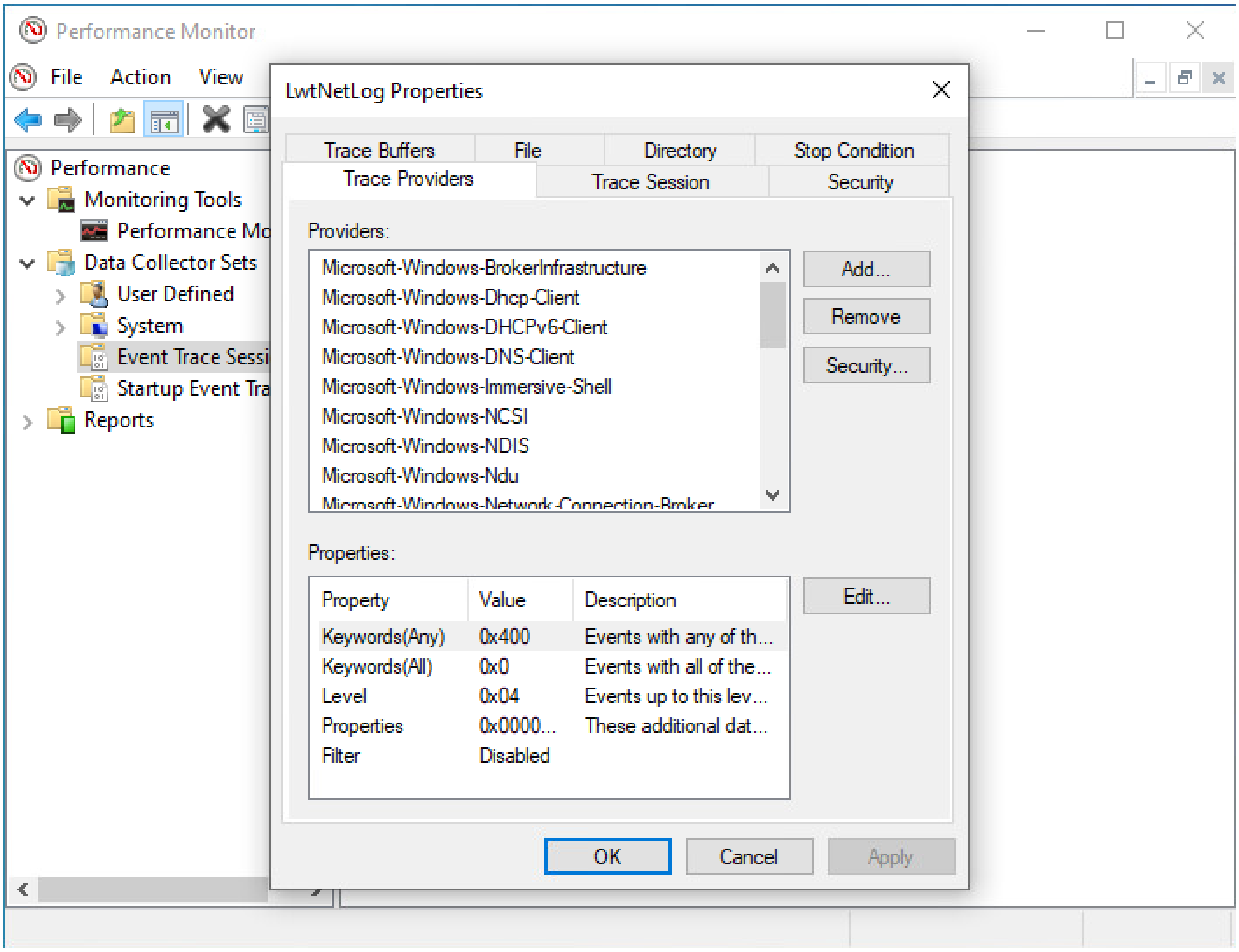


Figure 1: ETW architecture

- Provider: Applications and drivers that send events
- Consumer: Applications that receive events
- Session: Relays events sent from the provider, storing them in a buffer
- Controller: Creates, starts, and stops sessions (logman command[2]has controller functionality)

You can check ETW sessions from the Performance Monitor. It also allows you to create new sessions and prepare for event collection. As shown in Figure 2, multiple providers can be registered in a single session.



Providers:

Microsoft-Windows-BrokerInfrastructure
Microsoft-Windows-Dhcp-Client
Microsoft-Windows-DHCPv6-Client
Microsoft-Windows-DNS-Client
Microsoft-Windows-Immersive-Shell
Microsoft-Windows-NCSI
Microsoft-Windows-NDIS
Microsoft-Windows-Ndu
Microsoft-Windows-Network-Connection-Broker

Properties:

Property	Value	Description
Keywords(Any)	0x400	Events with any of th...
Keywords(All)	0x0	Events with all of the...
Level	0x04	Events up to this lev...
Properties	0x0000...	These additional dat...
Filter	Disabled	

Figure 2: Example of checking a session from Performance Monitor

You can also check which providers are registered on Windows OS by executing the following command. By default, more than 1,000 providers are registered.

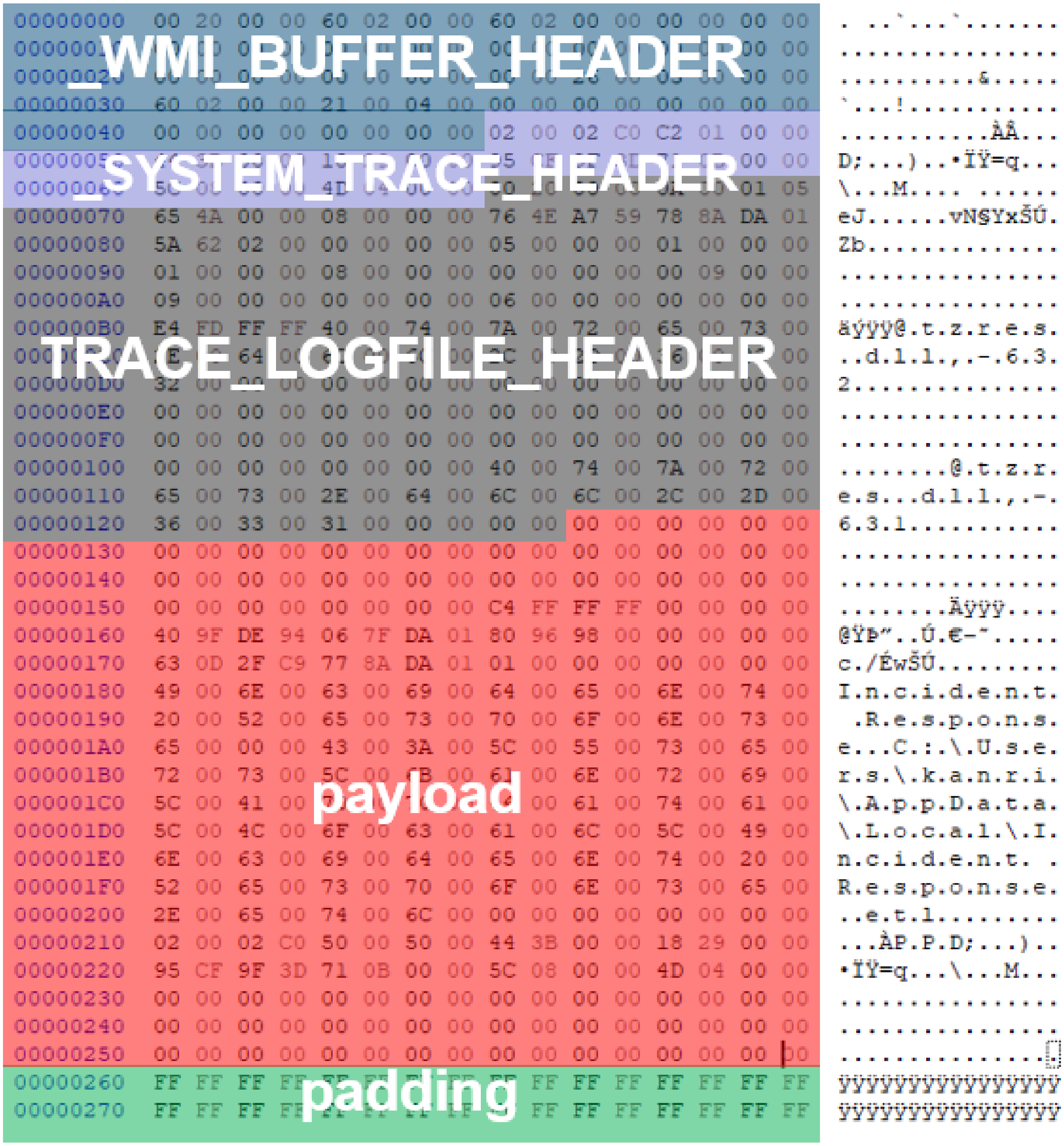
```
> logman query providers
```

With so many providers available by default, you probably thought that you would be able to collect various logs by using them. In particular, for the purposes of incident investigation and detecting suspicious behavior such as malware, the following providers would be useful.

- Microsoft-Windows-Threat-Intelligence: Detects behavior related to process injection, etc., which is used by malware.
- Microsoft-Windows-DNS-Client: Events related to name resolution
- Microsoft-Antimalware-AMFilter: Results of virus scans by Microsoft Defender
- Microsoft-Windows-Shell-Core: Events related to process execution and termination
- Microsoft-Windows-Kernel-Process: Events related to processes
- Microsoft-Windows-Kernel-File: Events related to file operations

ETW event format

There are two main ways for processing ETW events (Stream Mode). One of them is to save ETW events as an ETL file, and the other is to save ETW events in a buffer and receive them in real time. In both cases, ETW events are saved in the same format. Figure 3 shows the format of ETW events.

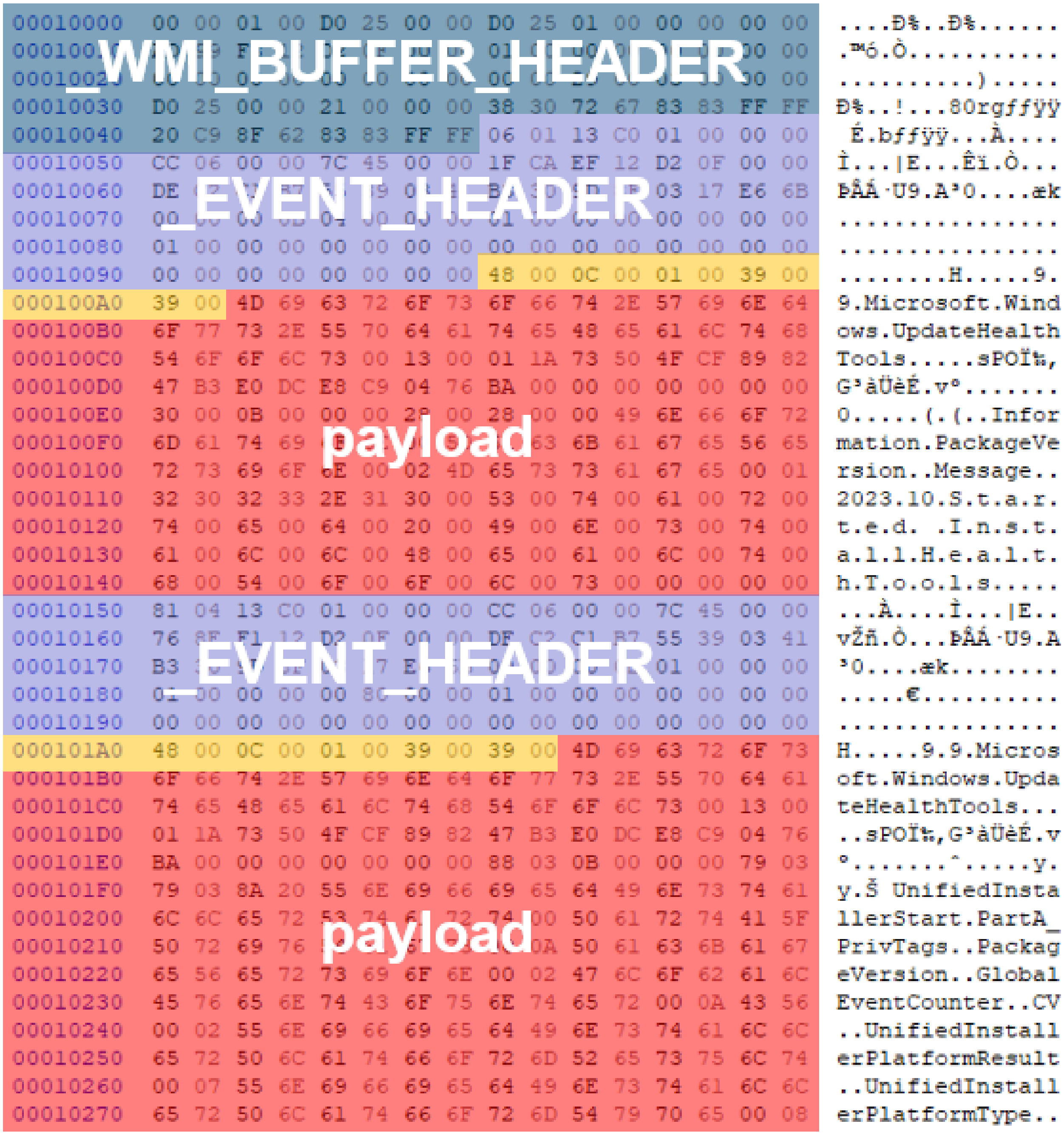


```
00000000  00 20 00 00 60 02 00 00 60 02 00 00 00 00 00 00  . ..`...`.......
00000010  [illegible] 00 00                                  ................
00000020  00 00 00 00 00 00 00 00 00 00 26 00 05 00 00 00  ..........&.....
00000030  60 02 00 00 21 00 04 00 00 00 00 00 00 00 00 00  `...!...........
00000040  00 00 00 00 00 00 00 00 02 00 02 C0 C2 01 00 00  ............ÀÂ..
00000050  44 3B 00 00 [illegible] 29 00 00 95 CF 9F 3D 71 [illegible] 00 00  D;...)..•ÏŸ=q...
00000060  5C 00 00 00 4D 04 00 00 [illegible] 20 [illegible] 01 05  \...M.... ......
00000070  65 4A 00 00 08 00 00 00 76 4E A7 59 78 8A DA 01  eJ......vN§YxŠÚ.
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  äýÿÿ@.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  6.3.1...........
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  ........Äÿÿÿ....
00000160  40 9F DE 94 06 7F DA 01 80 96 98 00 00 00 00 00  @ŸÞ”..Ú.€–˜.....
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  ...ÀP.P.D;...)..
00000220  95 CF 9F 3D 71 0B 00 00 5C 08 00 00 4D 04 00 00  •ÏŸ=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  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000270  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
```

Figure 3: ETW event format (the beginning of ETL file)

It starts with the _WMI_BUFFER_HEADER[3]. This header contains information such as the buffer size and offset, and the date and time the event was created. The next header depends on the contents that follow. In the case of an ETL file, the

_SYSTEM_TRACE_HEADER and _TRACE_LOGFILE_HEADER follow. If these headers are included, this indicates that it is the beginning of the ETL file and that no further ETW events are included. If ETW events are included, it will look like Figure 4.



```
00010000  00 00 01 00 D0 25 00 00 D0 25 01 00 00 00 00 00  ....Ð%..Ð%......
00010010  [illegible] 99 F[illegible] 01 [illegible] 00 00  .™ó.Ò...........
00010020  00 00 00 [illegible] 29 [illegible] 00 00  ...........).....
00010030  D0 25 00 00 21 00 00 00 38 30 72 67 83 83 FF FF  Ð%..!...80rgƒƒÿÿ
00010040  20 C9 8F 62 83 83 FF FF 06 01 13 C0 01 00 00 00   É.bƒƒÿÿ...À....
00010050  CC 06 00 00 7C 45 00 00 1F CA EF 12 D2 0F 00 00  Ì...|E...Êï.Ò...
00010060  DE [illegible] 03 17 E6 6B  ÞÂÁ·U9.A³0....æk
00010070  00 [illegible] 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.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.....sPOÏ‰‚
000100D0  47 B3 E0 DC E8 C9 04 76 BA 00 00 00 00 00 00 00  G³àÜèÉ.vº.......
000100E0  30 00 0B 00 00 00 28 00 28 00 00 49 6E 66 6F 72  0.....(.(..Infor
000100F0  6D 61 74 69 [illegible] 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  ...À....Ì...|E..
00010160  76 8E F1 12 D2 0F 00 00 DF C2 C1 B7 55 39 03 41  vŽñ.Ò...ßÂÁ·U9.A
00010170  B3 [illegible] 01 00 00 00  ³0....æk........
00010180  01 00 00 00 00 80 00 00 01 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  ..sPOÏ‰‚G³àÜèÉ.v
000101E0  BA 00 00 00 00 00 00 00 88 03 0B 00 00 00 79 03  º.......ˆ.....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 [illegible] 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..
```

Figure 4: ETW event format (ETW event)

The first part of the header still starts with _WMI_BUFFER_HEADER, but the next header is _EVENT_HEADER, followed by the actual event data.

It is difficult to parse ETW events manually because they have no signature and the type information contained in each header affects the headers that follow, as described above. On Windows OS, you can convert ETL files to EVTX files or CSV files as follows, because the tracerpt command is installed by default.

```
> tracerpt test.etl -o test.evtx -of EVTX -lr

> tracerpt test.etl -o test.csv -of CSV
```

ETW structure

You can check ETW configuration information to some extent using the performance monitor, logman command, and registry information introduced earlier. However, not all of the information can be checked using these methods, and you can also obtain various types of information from the ETW structure. However, it cannot be obtained in user mode, and so you will need to obtain it from kernel mode using a debugger or other method. You can trace the structure of ETW providers as shown in Figure 5.

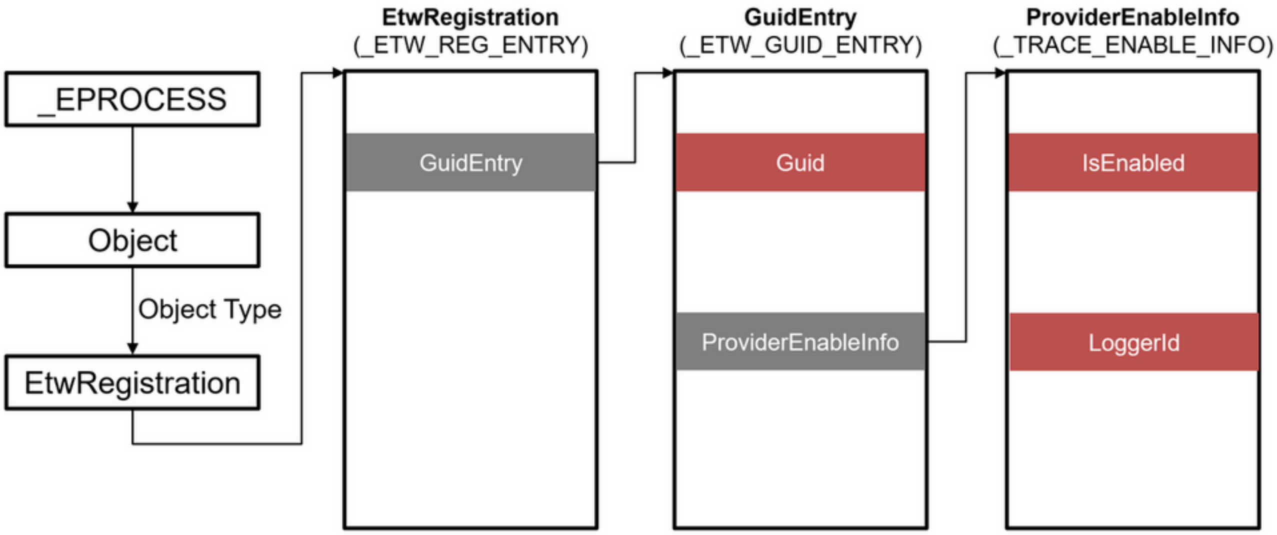


Figure 5: Structure of ETW providers

The structure of the ETW provider can be traced from an object with **EtwRegistration** object type in the process, and _ETW_GUID_ENTRY and _TRACE_ENABLE_INFO contain information such as GUID. Therefore, you can check which process is using which ETW provider. The structure of the ETW consumer can be traced as shown in Figure 6.

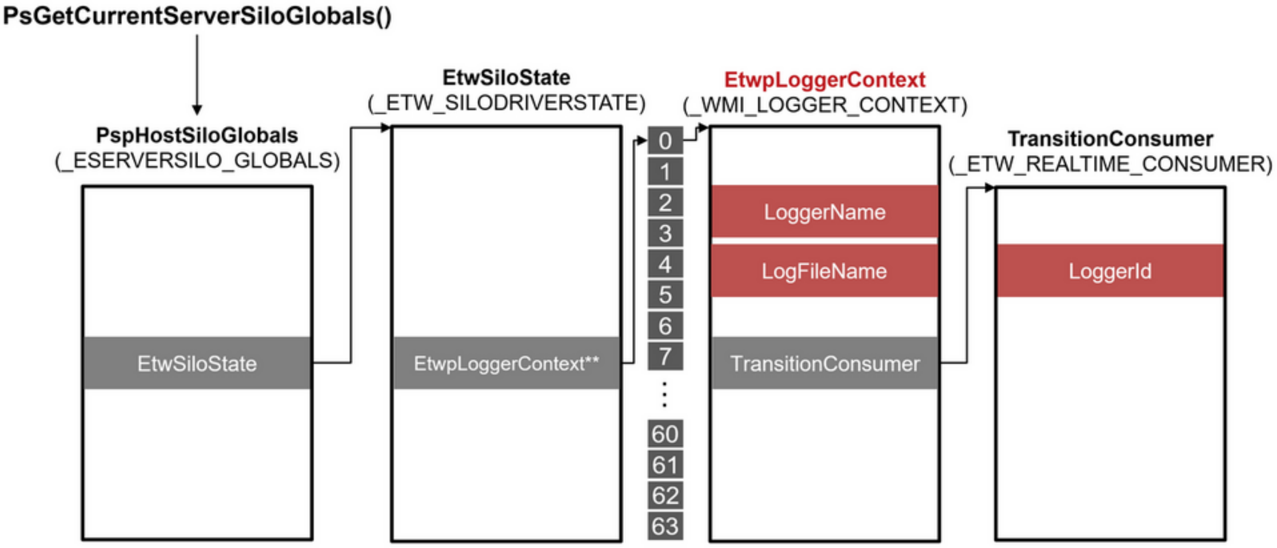


Figure 6: Structure of ETW consumers

You can trace the structure of the ETW consumer from the data obtained from the PsGetCurrentServerSiloGlobals function. _WMI_LOGGER_CONTEXT and _ETW_REALTIME_CONSUMER contain various information, and you can check the buffer size, current buffer usage, number of lost events, and more.

Recover ETW Events

Relations between ETW events and ETW structures

Some ETW events are saved as files by default, but in many cases, they are read from the buffer into the ETW consumer in real time, and so unless you configure them manually, most of them are not saved on the system as files. However, since ETW events are stored in the buffer, if you can collect the data, you may be able to use it for incident response or other purposes. Furthermore, even if the ETL file is deleted by the attacker, the ETW events may still be stored in the buffer.
As mentioned earlier, the ETW event format has no signature and cannot be recovered from disk or memory using file carving. For this reason, we explored methods to extract data from ETW structure.

As a result, we have identified the members of the structure that store ETW events as follows:

- GlobalList（_WMI_LOGGER_CONTEXT）
- BufferQueue（_WMI_LOGGER_CONTEXT）
- BatchedBufferList（_WMI_LOGGER_CONTEXT）
- CompressionTarget（_WMI_LOGGER_CONTEXT）
- UserBufferListHead（_ETW_REALTIME_CONSUMER）

GlobalList and BufferQueue are LIST_ENTRY, and the ETW events stored in the buffer are connected as a bi-directional linked list as shown in Figure 7. All the ETW events in the buffer are connected to GlobalList.

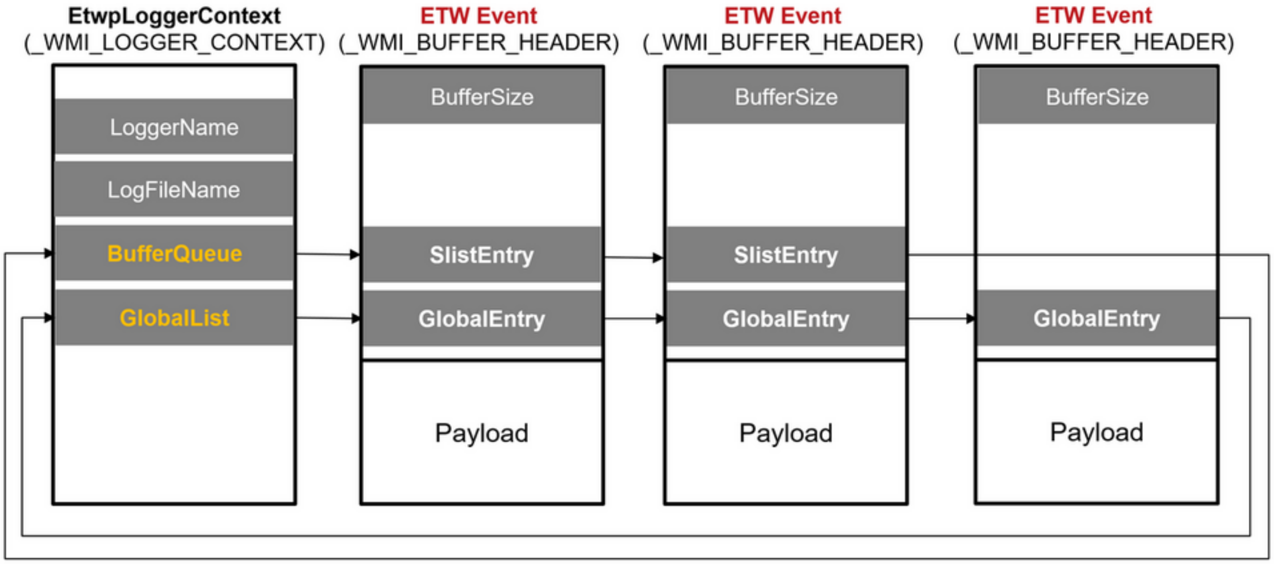


Figure 7: Relations between _WMI_LOGGER_CONTEXT and buffer

Because ETW structures are undocumented, it is not clear exactly why multiple members are related to the buffer in this way, but based on the behavior, it is possible that the ETW Stream Mode configuration affects it. Figure 8 shows the members considered to be related to each ETW Stream Mode. When it is set to save to an ETL file, BufferQueue is used, and when it is set to Real time, UserBufferListHead is used. Although there are differences in usage depending on the member, all ETW events are linked to GlobalList, and so it is probably best to refer to GlobalList when recovering ETW events.

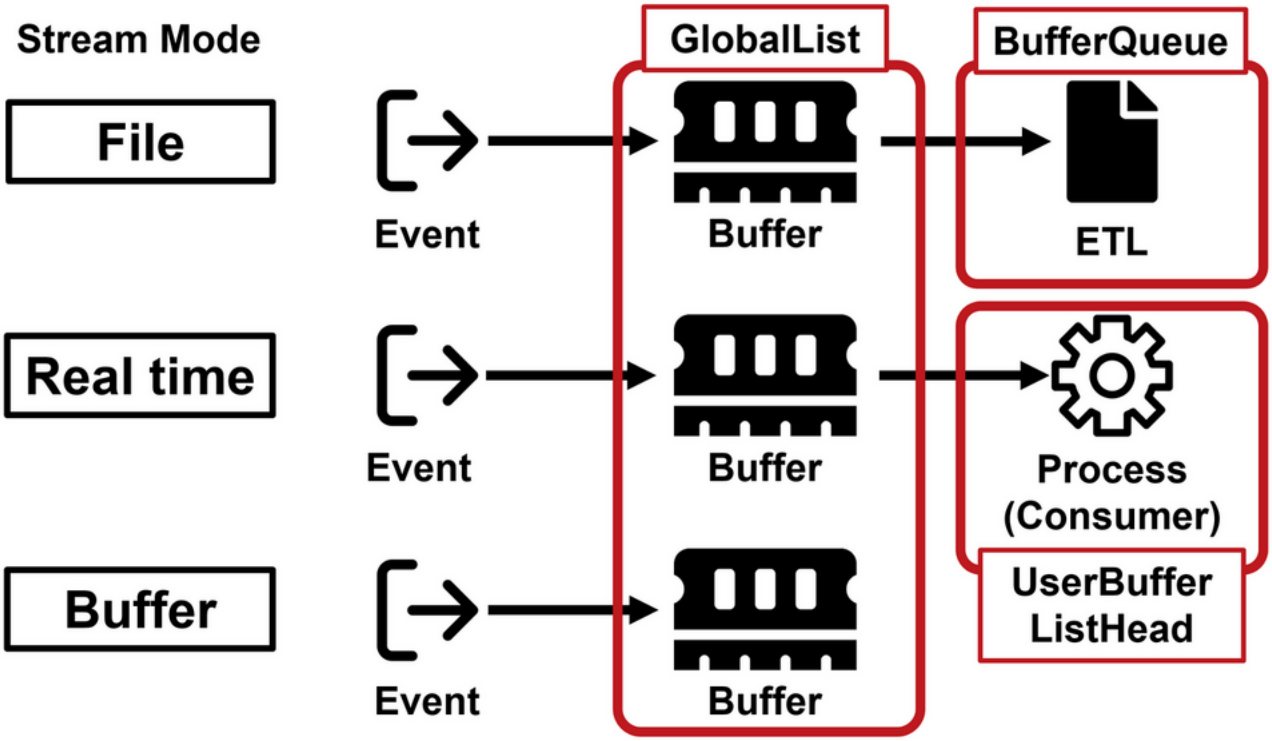
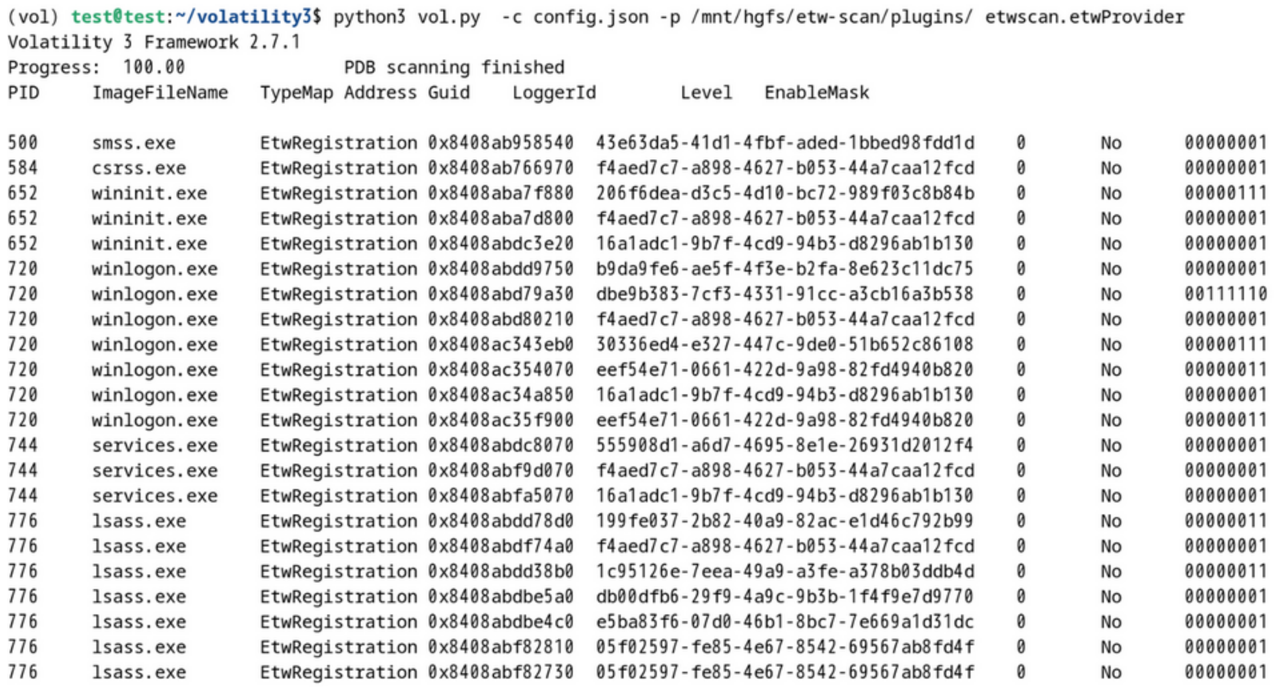


Figure 8: Relations between ETW Stream Mode and ETW structure members

ETW Scanner for Volatility3

Based on the above research results, we have created a tool for recovering ETW events from memory images. This is implemented as a plugin for The Volatility Framework (hereinafter referred to as "Volatility"), a memory forensics tool. Using this plugin, you can not only recover ETW events, but also check information about ETW providers and ETW consumers. Figure 9 shows an example of the plugin running.



```
(vol) test@test:~/volatility3$ python3 vol.py  -c config.json -p /mnt/hgfs/etw-scan/plugins/ etwscan.etwProvider
Volatility 3 Framework 2.7.1
Progress:  100.00               PDB scanning finished
PID     ImageFileName   TypeMap Address Guid    LoggerId        Level   EnableMask

500     smss.exe        EtwRegistration 0x8408ab958540  43e63da5-41d1-4fbf-aded-1bbed98fdd1d    0       No      00000001
584     csrss.exe       EtwRegistration 0x8408ab766970  f4aed7c7-a898-4627-b053-44a7caa12fcd    0       No      00000001
652     wininit.exe     EtwRegistration 0x8408aba7f880  206f6dea-d3c5-4d10-bc72-989f03c8b84b    0       No      00000111
652     wininit.exe     EtwRegistration 0x8408aba7d800  f4aed7c7-a898-4627-b053-44a7caa12fcd    0       No      00000001
652     wininit.exe     EtwRegistration 0x8408abdc3e20  16a1adc1-9b7f-4cd9-94b3-d8296ab1b130    0       No      00000001
720     winlogon.exe    EtwRegistration 0x8408abdd9750  b9da9fe6-ae5f-4f3e-b2fa-8e623c11dc75    0       No      00000001
720     winlogon.exe    EtwRegistration 0x8408abd79a30  dbe9b383-7cf3-4331-91cc-a3cb16a3b538    0       No      00111110
720     winlogon.exe    EtwRegistration 0x8408abd80210  f4aed7c7-a898-4627-b053-44a7caa12fcd    0       No      00000001
720     winlogon.exe    EtwRegistration 0x8408ac343eb0  30336ed4-e327-447c-9de0-51b652c86108    0       No      00000111
720     winlogon.exe    EtwRegistration 0x8408ac354070  eef54e71-0661-422d-9a98-82fd4940b820    0       No      00000011
720     winlogon.exe    EtwRegistration 0x8408ac34a850  16a1adc1-9b7f-4cd9-94b3-d8296ab1b130    0       No      00000001
720     winlogon.exe    EtwRegistration 0x8408ac35f900  eef54e71-0661-422d-9a98-82fd4940b820    0       No      00000011
744     services.exe    EtwRegistration 0x8408abdc8070  555908d1-a6d7-4695-8e1e-26931d2012f4    0       No      00000001
744     services.exe    EtwRegistration 0x8408abf9d070  f4aed7c7-a898-4627-b053-44a7caa12fcd    0       No      00000001
744     services.exe    EtwRegistration 0x8408abfa5070  16a1adc1-9b7f-4cd9-94b3-d8296ab1b130    0       No      00000001
776     lsass.exe       EtwRegistration 0x8408abdd78d0  199fe037-2b82-40a9-82ac-e1d46c792b99    0       No      00000011
776     lsass.exe       EtwRegistration 0x8408abdf74a0  f4aed7c7-a898-4627-b053-44a7caa12fcd    0       No      00000001
776     lsass.exe       EtwRegistration 0x8408abdd38b0  1c95126e-7eea-49a9-a3fe-a378b03ddb4d    0       No      00000011
776     lsass.exe       EtwRegistration 0x8408abdbe5a0  db00dfb6-29f9-4a9c-9b3b-1f4f9e7d9770    0       No      00000001
776     lsass.exe       EtwRegistration 0x8408abdbe4c0  e5ba83f6-07d0-46b1-8bc7-7e669a1d31dc    0       No      00000001
776     lsass.exe       EtwRegistration 0x8408abf82810  05f02597-fe85-4e67-8542-69567ab8fd4f    0       No      00000001
776     lsass.exe       EtwRegistration 0x8408abf82730  05f02597-fe85-4e67-8542-69567ab8fd4f    0       No      00000001
```

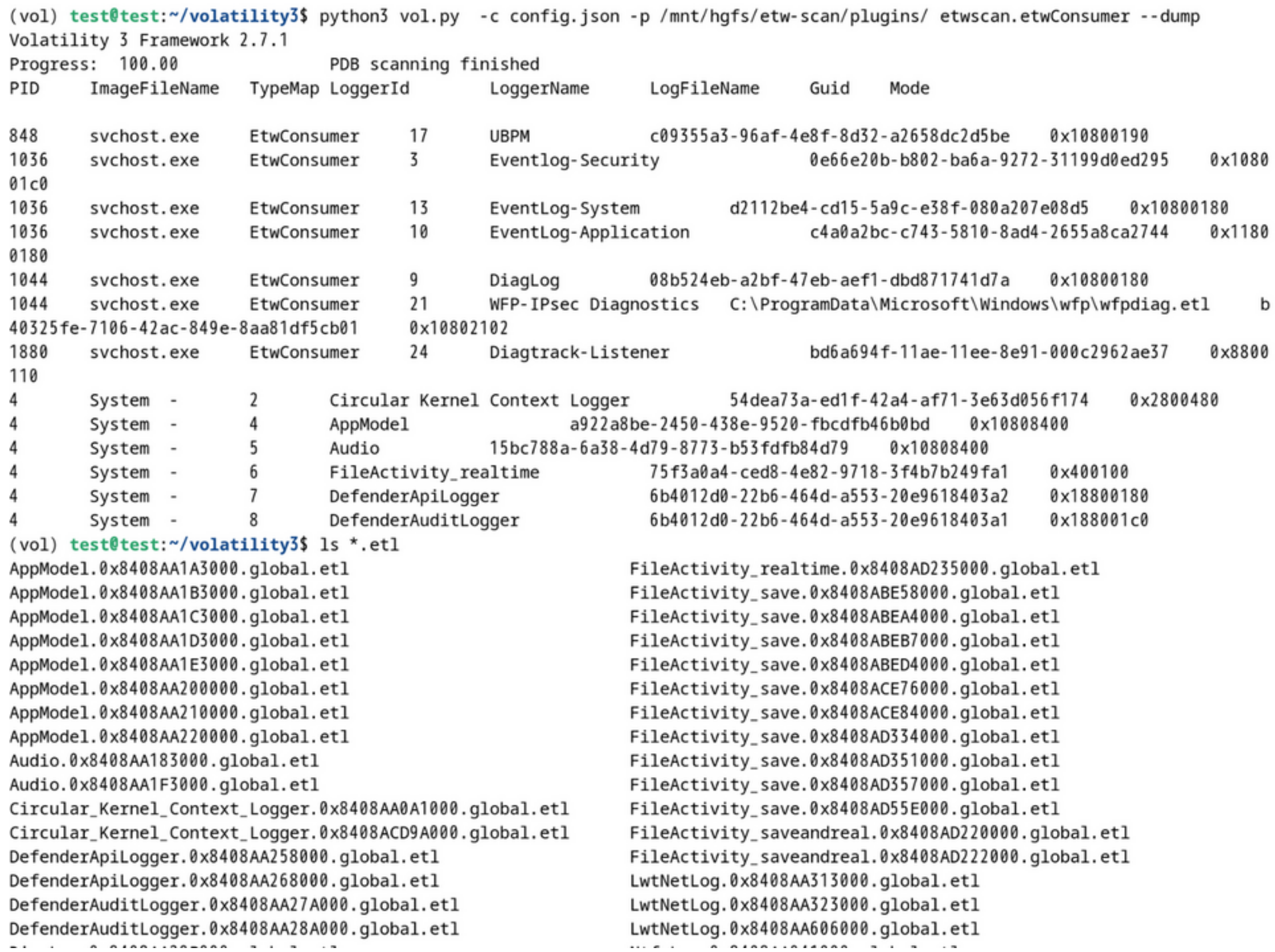
Figure 9: Example of executing a plugin

You can download this plugin from the following GitHub repository. We hope you find it useful.

GitHub: JPCERTCC/etw-scan
https://github.com/JPCERTCC/etw-scan

Using the recovered ETW event in incident investigations

Now, let’s look at some examples of how to use the recovered ETW events in incident investigations. To recover ETW events, specify the option **--dump** (for GlobalList only) or **--alldump** (for all members) as follows. The number of ETW events that can be recovered depends on the environment, but as shown in Figure 10, it is possible to recover a large number of ETW events as ETL files.

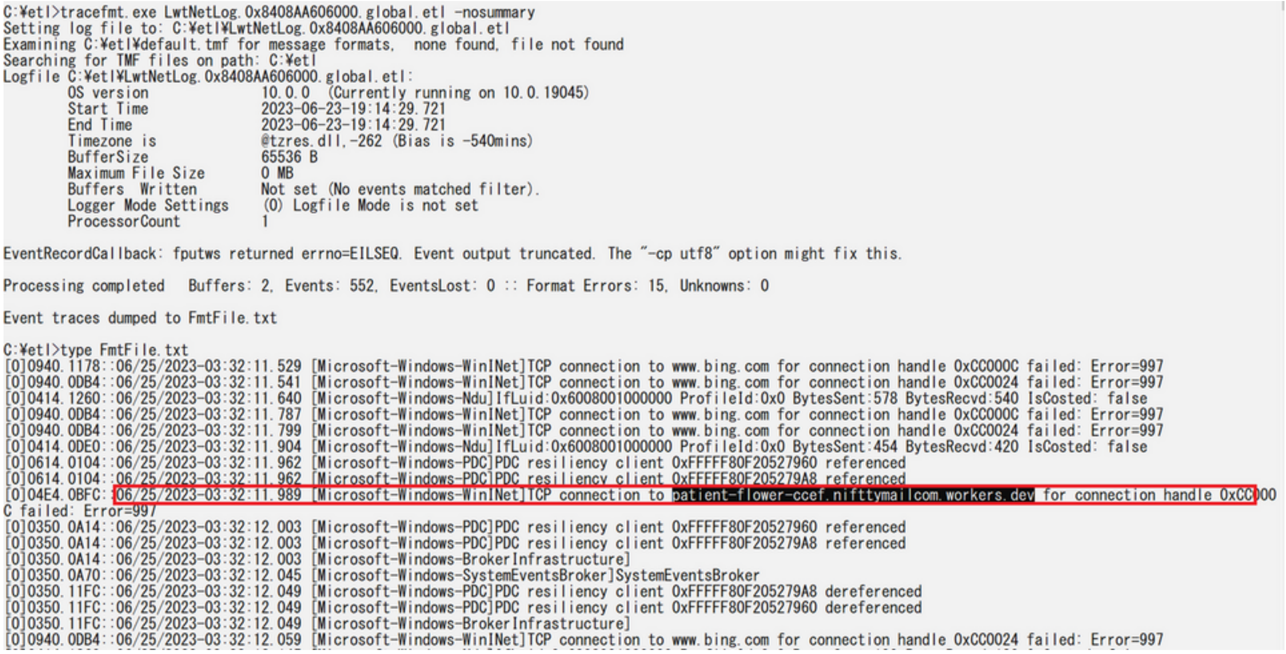


```
(vol) test@test:~/volatility3$ python3 vol.py  -c config.json -p /mnt/hgfs/etw-scan/plugins/ etwscan.etwConsumer --dump
Volatility 3 Framework 2.7.1
Progress:  100.00               PDB scanning finished
PID     ImageFileName   TypeMap LoggerId        LoggerName      LogFileName     Guid    Mode

848     svchost.exe     EtwConsumer     17      UBPM            c09355a3-96af-4e8f-8d32-a2658dc2d5be    0x10800190
1036    svchost.exe     EtwConsumer     3       Eventlog-Security               0e66e20b-b802-ba6a-9272-31199d0ed295    0x1080
01c0
1036    svchost.exe     EtwConsumer     13      EventLog-System         d2112be4-cd15-5a9c-e38f-080a207e08d5    0x10800180
1036    svchost.exe     EtwConsumer     10      EventLog-Application            c4a0a2bc-c743-5810-8ad4-2655a8ca2744    0x1180
0180
1044    svchost.exe     EtwConsumer     9       DiagLog         08b524eb-a2bf-47eb-aef1-dbd871741d7a    0x10800180
1044    svchost.exe     EtwConsumer     21      WFP-IPsec Diagnostics   C:\ProgramData\Microsoft\Windows\wfp\wfpdiag.etl     b
40325fe-7106-42ac-849e-8aa81df5cb01     0x10802102
1880    svchost.exe     EtwConsumer     24      Diagtrack-Listener              bd6a694f-11ae-11ee-8e91-000c2962ae37    0x8800
110
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
(vol) test@test:~/volatility3$ ls *.etl
AppModel.0x8408AA1A3000.global.etl                          FileActivity_realtime.0x8408AD235000.global.etl
AppModel.0x8408AA1B3000.global.etl                          FileActivity_save.0x8408ABE58000.global.etl
AppModel.0x8408AA1C3000.global.etl                          FileActivity_save.0x8408ABEA4000.global.etl
AppModel.0x8408AA1D3000.global.etl                          FileActivity_save.0x8408ABEB7000.global.etl
AppModel.0x8408AA1E3000.global.etl                          FileActivity_save.0x8408ABED4000.global.etl
AppModel.0x8408AA200000.global.etl                          FileActivity_save.0x8408ACE76000.global.etl
AppModel.0x8408AA210000.global.etl                          FileActivity_save.0x8408ACE84000.global.etl
AppModel.0x8408AA220000.global.etl                          FileActivity_save.0x8408AD334000.global.etl
Audio.0x8408AA183000.global.etl                             FileActivity_save.0x8408AD351000.global.etl
Audio.0x8408AA1F3000.global.etl                             FileActivity_save.0x8408AD357000.global.etl
Circular_Kernel_Context_Logger.0x8408AA0A1000.global.etl    FileActivity_save.0x8408AD55E000.global.etl
Circular_Kernel_Context_Logger.0x8408ACD9A000.global.etl    FileActivity_saveandreal.0x8408AD220000.global.etl
DefenderApiLogger.0x8408AA258000.global.etl                 FileActivity_saveandreal.0x8408AD222000.global.etl
DefenderApiLogger.0x8408AA268000.global.etl                 LwtNetLog.0x8408AA313000.global.etl
DefenderAuditLogger.0x8408AA27A000.global.etl               LwtNetLog.0x8408AA323000.global.etl
DefenderAuditLogger.0x8408AA28A000.global.etl               LwtNetLog.0x8408AA606000.global.etl
```

Figure 10: Example of recovering ETW events

You can parse the recovered ETL file and check for important information. For example, there is an ETW session called LwtNetLog that is enabled by default. This ETW session has multiple network-related ETW providers configured, and it collects various types of information, including communication packets, DNS access, and DHCP. Check the recovered ETW events, and you can see the destination where the malware communicates, as shown in Figure 11. To parse the ETL file, we used tracefmt[4] This tool is not installed by default, and so you will need to install it manually.



```
C:¥etl>tracefmt.exe LwtNetLog.0x8408AA606000.global.etl -nosummary
Setting log file to: C:¥etl¥LwtNetLog.0x8408AA606000.global.etl
Examining C:¥etl¥default.tmf for message formats,  none found, file not found
Searching for TMF files on path: C:¥etl
Logfile C:¥etl¥LwtNetLog.0x8408AA606000.global.etl:
        OS version              10.0.0  (Currently running on 10.0.19045)
        Start Time              2023-06-23-19:14:29.721
        End Time                2023-06-23-19:14:29.721
        Timezone is             @tzres.dll,-262 (Bias is -540mins)
        BufferSize              65536 B
        Maximum File Size       0 MB
        Buffers  Written        Not set (No events matched filter).
        Logger Mode Settings    (0) Logfile Mode is not set
        ProcessorCount          1

EventRecordCallback: fputws returned errno=EILSEQ. Event output truncated. The "-cp utf8" option might fix this.

Processing completed   Buffers: 2, Events: 552, EventsLost: 0 :: Format Errors: 15, Unknowns: 0

Event traces dumped to FmtFile.txt

C:¥etl>type FmtFile.txt
[0]0940.1178::06/25/2023-03:32:11.529 [Microsoft-Windows-WinINet]TCP connection to www.bing.com for connection handle 0xCC000C failed: Error=997
[0]0940.0DB4::06/25/2023-03:32:11.541 [Microsoft-Windows-WinINet]TCP connection to www.bing.com for connection handle 0xCC0024 failed: Error=997
[0]0414.1260::06/25/2023-03:32:11.640 [Microsoft-Windows-Ndu]IfLuid:0x6008001000000 ProfileId:0x0 BytesSent:578 BytesRecvd:540 IsCosted: false
[0]0940.0DB4::06/25/2023-03:32:11.787 [Microsoft-Windows-WinINet]TCP connection to www.bing.com for connection handle 0xCC000C failed: Error=997
[0]0940.0DB4::06/25/2023-03:32:11.799 [Microsoft-Windows-WinINet]TCP connection to www.bing.com for connection handle 0xCC0024 failed: Error=997
[0]0414.0DE0::06/25/2023-03:32:11.904 [Microsoft-Windows-Ndu]IfLuid:0x6008001000000 ProfileId:0x0 BytesSent:454 BytesRecvd:420 IsCosted: false
[0]0614.0104::06/25/2023-03:32:11.962 [Microsoft-Windows-PDC]PDC resiliency client 0xFFFFF80F20527960 referenced
[0]0614.0104::06/25/2023-03:32:11.962 [Microsoft-Windows-PDC]PDC resiliency client 0xFFFFF80F205279A8 referenced
[0]04E4.0BFC::06/25/2023-03:32:11.989 [Microsoft-Windows-WinINet]TCP connection to patient-flower-ccef.nifttymailcom.workers.dev for connection handle 0xCC000
C failed: Error=997
[0]0350.0A14::06/25/2023-03:32:12.003 [Microsoft-Windows-PDC]PDC resiliency client 0xFFFFF80F20527960 referenced
[0]0350.0A14::06/25/2023-03:32:12.003 [Microsoft-Windows-PDC]PDC resiliency client 0xFFFFF80F205279A8 referenced
[0]0350.0A14::06/25/2023-03:32:12.003 [Microsoft-Windows-BrokerInfrastructure]
[0]0350.0A70::06/25/2023-03:32:12.045 [Microsoft-Windows-SystemEventsBroker]SystemEventsBroker
[0]0350.11FC::06/25/2023-03:32:12.049 [Microsoft-Windows-PDC]PDC resiliency client 0xFFFFF80F205279A8 dereferenced
[0]0350.11FC::06/25/2023-03:32:12.049 [Microsoft-Windows-PDC]PDC resiliency client 0xFFFFF80F20527960 dereferenced
[0]0350.11FC::06/25/2023-03:32:12.049 [Microsoft-Windows-BrokerInfrastructure]
[0]0940.0DB4::06/25/2023-03:32:12.059 [Microsoft-Windows-WinINet]TCP connection to www.bing.com for connection handle 0xCC0024 failed: Error=997
```

Figure 11: Checking the recovered LwtNetLog session

Furthermore, if EDR or antivirus software is installed, you may be able to recover the ETW events that these applications were trying to collect. Since each application tries to collect data from different ETW providers, there may be some differences, but still there is a possibility that useful ETW events such as Microsoft-Windows-Threat-Intelligence are recovered.

In closing

On Windows OS, it is possible to collect various information using ETW by default. Although we did not introduce it this time, it is also possible to monitor the system by creating a simple EDR that combines the information collection capabilities of ETW with detection logic. You can try using ETW for system monitoring and incident response.

Shusei Tomonaga
(Translated by Takumi Nakano)

References

[1] Microsoft: Event trace
https://learn.microsoft.com/en-us/windows/win32/etw/about-event-tracing

[2] Microsoft: logman
https://learn.microsoft.com/en-us/windows-server/administration/windows-commands/logman

[3] Geoff Chappell, Software Analyst: Kernel-Mode Windows
https://www.geoffchappell.com/studies/windows/km/ntoskrnl/api/index.htm

[4] Microsoft: Tracefmt
https://learn.microsoft.com/en-us/windows-hardware/drivers/devtest/tracefmt