Event Tracing for Windows (ETW) logs kernel, application and other system activity. ETW provides better data and uses less resources. By understanding the key characteristics of ETW, system administrators can make a well informed decision on how to utilize the logs collected via ETW to improve IT Security.

About Event Tracing for Windows

Event Tracing for Windows is a kernel-level tracing facility provided by the operating system and enabled dynamically. It logs kernel, application, and other system events.

Available since Windows 2000, ETW provides more detailed information on the operating system environment and application interaction than other logging services on Windows. In addition, ETW does this with less overhead and higher efficiency.

The architecture of ETW is straightforward, an event provider (any user-mode application, managed application or driver) writes events to ETW sessions. When events are written, ETW adds supplementary information about the event time, the process and thread ID that generated the event, the processor ID and the CPU usage data of the logging thread. This information is then ingested by event consumers, either from log files and/or by listening to tracing sessions in real time. Consumers then continue with any other configured processing activities.

The following three, independently functioning component types determine what is logged, when it is logged and where the log events are collected, all with relatively little system overhead. Together these components define an event tracing session.

Controllers

Controllers enable providers to log events to a session. They start, stop and define event trace sessions as well as specify the session/log file name, location, type and define the way of resolving date-time stamps.

Providers

Providers are applications equipped with event tracing instrumentation. When they are enabled by a controller, they send log events to a consumer.

Consumers

Consumers consume events from one or more event tracing sessions and retrieve events stored in log files along with logs from other real-time sessions. In this context, the log collector(s) act as consumers, ingesting generated events from enabled providers.
Apart from regular event tracing sessions, special purpose tracing sessions are also available, such as Private Logger, System Trace, NT Kernel Logger, AutoLogger and Global Logger. These sessions have predefined data and location settings and often provide the only way to access certain data, such as when events are created early in the boot process of a system.

## The Event Trace Data

ETW events include rich metadata, localizable message strings and schematized data payloads. They can be read from memory or saved to a file after decoding it on the source system. The Event Trace Log (ETL) file has a binary format and its content must also be decoded for viewing. Windows provides a GUI (Event Viewer), and a command line utility (tracerpt) for this purpose as well as other event tracing tools such as logman.

Below is an excerpt from an example of an event trace in XML format, exported from an ETL file.

### Event trace in XML format

```xml
<EventData>
  <Data Name="BufferSize">16384</Data>
  <Data Name="Version">83951626</Data>
  <Data Name="ProviderVersion">14393</Data>
  <Data Name="NumberOfProcessors">1</Data>
  <Data Name="MaxFileSize">2</Data>
  <Data Name="LogFileMode">8x82</Data>
  <Data Name="BuffersWritten">3</Data>
  <Data Name="StartBuffers">1</Data>
  <Data Name="PointerSize">8</Data>
  <Data Name="CPUSpeed">2400</Data>
  <Data Name="SessionNameString">WdiContextLog</Data>
  <Data Name="LogFileNameString">C:\Windows\System32\WDI\LogFiles\WdiContextLog.etl.002</Data>
</EventData>
```

**NOTE**

The data availability and the functional particulars of an event session may vary based on the operating system version. Please refer to the Event Tracing for Windows documentation for details.

## The Advantages of ETW

ETW has matured enough to become a well established and useful technology by now. Below are some of the key points in favor of ETW.

### Provides More In-Depth Data

ETW provides more thorough, more detailed and more timely data than its log management
predecessors on Windows. Since the release of Windows Vista, Windows Event Log and ETW use the same API to log events, allowing administrators to directly access data previously collected via Event Logging and to data that was previously unavailable.

For example: the *Microsoft-Windows-PowerShell* ETW Provider records which command was run as part of its payload, whereas Windows Event Log merely shows that PowerShell was invoked.

**Widens Log Data Reach**

Often, ETW is the only way to access certain log data. DNS logs, for instance, are captured from the *Microsoft-Windows-DNSServer* ETW provider by enabling Analytical Logging. This is the preferred method for collecting logs from Windows Server versions 2012 R2 (with hotfix 2956577) and later, but this method is not available in earlier versions of Windows Server.

**Consumes Less System Resources**

ETW uses kernel-level providers and manages memory buffers per session, consuming, as a rule, considerably fewer resources than other logging technologies on Windows, which is especially important when deployed in production environments.

**Built up of Independent Components**

ETW allows the dynamic updating of sessions and as the components (controllers, providers and consumers) function independently, they can be stopped, started and updated without rebooting the system or restarting the application(s) of interest. Troubleshooting live environments is practically unthinkable without the flexibility it enables.

**The Challenges of ETW**

There are all good reasons to use ETW, but it also introduces some challenges for system administrators to consider.

**Technology Changes**

ETW has undergone substantial changes with each major release since its introduction in 2000.

The updates added new functions and data structures, typically available only in the updated version of the operating system. An excellent example for this is the format of log files: the original Managed Object Format (MOF) was mostly superseded by the Trace Message Format (TMF), which in turn gave way to the manifest-based XML format. Some earlier components even required a separate PDB (program debugging database) to decode events. The newest format, TraceLogging (TL), was introduced with Windows 10 and embeds decoding information with the recorded log data.

It is clear that managing log collection across multiple file formats increases the overhead on system administrators.
Tooling Complexity

During the lifespan of ETW, tools for collecting, inspecting, analyzing and processing event tracing data have proliferated to keep up with the evolving functional and structural changes introduced.

Finding a solution to structure and centralize the data used by these tools is a must to enable the execution of security analysis tasks such as pattern matching and correlation. However, due to the limited GUI availability and inconsistent CLI tooling options, the burden of managing ETW data can quickly become time-consuming and expensive.

Capturing Issues

ETW is a “best effort” framework and does not guarantee that all events are captured. Any factor, that could potentially reduce the logging performance of a given system, such as the availability of memory and reduced storage responsiveness is a potential source of problem for reliably capturing event traces.

ETW in Real World Threat Hunting

Administrators are already aware, that attack tactics to cover malicious tracks, such as the Windows Event Log being cleared, will emit its own telemetry in the form of additional Event IDs. By knowing this, administrators are proactively monitoring and alerting for these Event IDs. However, these tactics can fall short if attackers start tampering with ETW, as it can be used as a mechanism to interfere with the flow of logging.

A technique covered by Palantir involves the removing of an ETW Provider from a trace session. As a result, the targeted Event Log is no longer supplied with the correct system telemetry. For administrators, that are reliant on Windows Event Log-based threat detection only, this trick will likely to go unnoticed.

There are a few options to mitigate the effects of tampering with ETW, as this technique can be detected early, if there is a centralized log collection strategy in place to collect logs from the relevant ETW Provider. By collecting event traces with the NXLog Event Tracing for Windows (im_etw) module, administrators can collect and forward traces to a third party suite, such as a SIEM, for further monitoring and incident response.

Other cases where ETW is utilized for attack detection and defense operations include Domain Name System (DNS) log collection to help mitigate attacks involving DNS. Collecting ETW traces also further informs threat detection and operational support in collecting telemetry from critical applications such as Windows Firewall.
Using NXLog as a Single Agent Solution to Collect ETW Logs

To harvest the benefits and overcome the challenges of working with ETW, a single log collection agent is required that poses the following:

- Act both as a **Controller and Consumer**, therefore being able to start the tracing session and collect events directly from the Consumer.
- Capable of collecting ETW trace data and then forward it **without saving the data to disk**, and is able to do it with high efficiency.
- Incorporates the technology to **enrich the raw data** once ETW data is collected, by being able to parse, format and convert the logs for further processing.
- Includes the mechanisms to forward the enriched **raw data** to the required nodes by helping to ease system disk load, in order to avoid the losing of event traces due to the inability to write and process all traces that are being generated.

**NXLog does it all using configurable modules.**

There are solutions that rely on Microsoft provided tools to capture the ETW trace to disk before decoding the trace file into a human readable format for further processing and analysis. These methods are inefficient and unreliable as they consume system resources in both the storage requirements capturing the trace to disk and in processing or decoding the trace file.

The **Event Tracing for Windows (im_etw) module** in the NXLog Enterprise Edition allows administrators to use just one log collection software for all their ETW log collection requirements. The agent will collect ETW data at the source, then further parse, format, and forward the data as needed.

With structured log enrichment and centralized log collection, administrators can easily correlate data, create patterns and visualize event data provided by the sources. They can utilize their already established infrastructure such as SIEMs, log management suites, and other dashboards - for data analysis, reporting, alerting, and incident response based on the insights provided by NXLog.

---

**NXLog Ltd. develops multi-platform log collection tools that support many different log sources, formats, transports, and integrations. The tools help administrators collect, parse, and forward logs so they can more easily respond to security issues, investigate operational problems, and analyze event data. NXLog distributes the free and open source **NXLog Community Edition** and offers additional features and support with the **NXLog Enterprise Edition**.**

*This document is provided for informational purposes only and is subject to change without notice.*