Original Article
ISSN (Online): 2582-7472

ShodhKosh: Journal of Visual and Performing Arts
January-June 2024 5(1), 3250–3256

A FORENSIC PERSPECTIVE ON THE USE OF EVENT VIEWER FOR DETECTING MALICIOUS ACTIVITIES AND ENSURING SYSTEM INTEGRITY

Premal C. Patel [1], Pina M. Bhatt [2], Umang Parmar [3], Keval Bhavsar [3]

[1] Department of Computer Engineering, College of Technology, Silver Oak University, Ahmedabad, Gujarat - 382481, India
[2] Department of Mechanical Engineering, College of Technology, Silver Oak University, Ahmedabad, Gujarat - 382481, India
[3] Department of Mechanical Engineering, Aditya Silver Oak Institute of Technology, Silver Oak University, Ahmedabad, Gujarat - 382481, India

ABSTRACT

Event Viewer is a vital tool embedded within Microsoft Windows that records a wide range of system, security, and application-related events. For forensic investigators, these logs are crucial in identifying signs of malicious activities, reconstructing timelines, and maintaining system integrity. This paper highlights the role of Event Viewer in digital forensics, discussing how specific logs from various categories—Application, Security, Setup, System, and Forwarded Events—can be extracted, parsed, and stored in XML format for in-depth analysis. Furthermore, the paper proposes a structured XML-based data model for efficient forensic storage and analysis, compares it with other log management approaches, and demonstrates its effectiveness in digital investigations.

Keywords: Digital Forensics, Event Viewer, Windows Logs, XML Storage, System Integrity, Malware Detection, Log Analysis, Timeline Reconstruction, Cyber-Security, Incident Response

Corresponding Author

Pina M. Bhatt, pmbhatt15@gmail.com

DOI
10.29121/shodhkosh.v5.i1.2024.5975

Funding: This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

1. INTRODUCTION

Event Logs are commonly analyzed during incident investigations—especially in cases involving malware infections—to trace events that might reveal the nature or source of the incident. However, it's important to understand that Windows Event Logs were not specifically designed to detect suspicious or malicious behavior. As a result, they often lack the depth or granularity required for comprehensive forensic analysis.

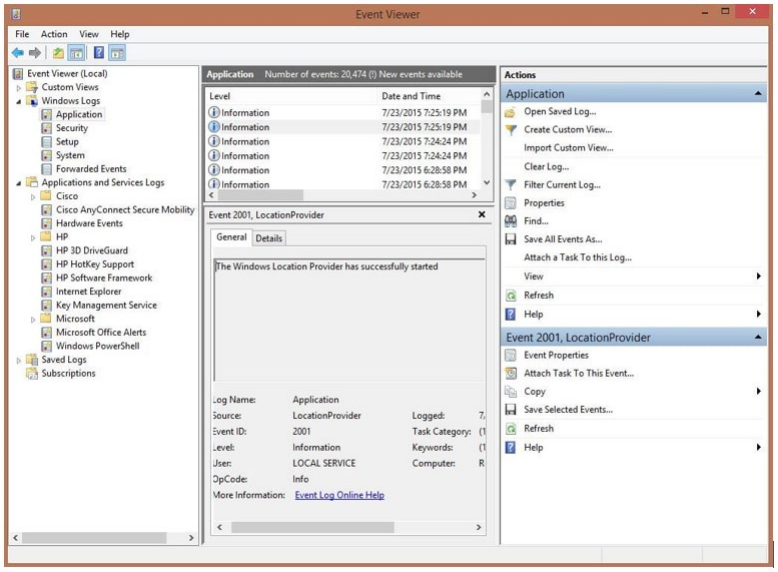

Image1 Logs of Event viewer

Event Viewer is a built-in Windows utility that maintains logs of various system-level and user-level events. It serves as a critical source of evidence in digital forensic investigations. The tool categorizes events into distinct logs:

- Q1 (Application Logs): Logs generated by installed applications and software behavior.
- Q2 (Security Logs): Logs related to login attempts, access controls, and audit policies.
- Q3 (Setup Logs): Logs created during installation or setup of applications and system components.
- Q4 (System Logs): Logs produced by Windows system components and drivers.
- Q5 (Forwarded Events): Logs forwarded from other systems via event subscriptions

For forensic analysis, extracting metadata from each of these categories and converting it into a structured XML format allows for better querying, storage, and comparison.

2. XML METADATA STORAGE STRUCTURE (Q1–Q5)

```
<EventLog>
<ApplicationLogs id="Q1">
<Event>
<EventID>1000</EventID>
<Source>Application Error</Source>
<TimeCreated>2023-07-24T08:30:00</TimeCreated>
<User>SYSTEM</User>
<Level>Error</Level>
<Message>Faulting application path...</Message>
</Event>
</ApplicationLogs>
<SecurityLogs id="Q2">
<Event>
<EventID>4625</EventID>
<Source>Microsoft Windows security auditing</Source>
<TimeCreated>2023-07-24T08:45:00</TimeCreated>
<User>unknown</User>
<Level>Warning</Level>
<Message>Failed logon attempt...</Message>
```

```
</Event>
</SecurityLogs>
<!-- SetupLogs Q3, SystemLogs Q4, ForwardedEvents Q5 follow similar structure -->
</EventLog>
```

3. LITERATURE REVIEW

The use of system and application logs in forensic investigations has been explored by several researchers and practitioners. Logs provide a timeline of events and can act as a trustworthy data source for identifying security breaches.

Garfinkel [1] proposed digital forensic XML structures for long-term log storage and metadata analysis. His work emphasized the need for schema-driven log management that could be directly integrated with forensic tools.

There are Some features of Event Tracing for Windows (ETB) configuration can be observed using other tools for Performance Monitor, the logman command, or by reviewing the relevant registry entries in the system. These methods define the basic concepts of ETW configured by active providers. They do not provide complete visibility into the internal workings of the ETW framework.

To get more celerity like detailed structural information about ETW providers and its behavior we need to explore beyond the user mode. This level of access is requires working in kernel mode by using specialized tools like a kernel debugger.

By analyzing the ETW structures at the kernel level, it is possible to trace the architecture and interactions of ETW providers more accurately. Figure [X] illustrates how this tracing process can be performed to reveal the relationships and flow of events within the ETW framework.

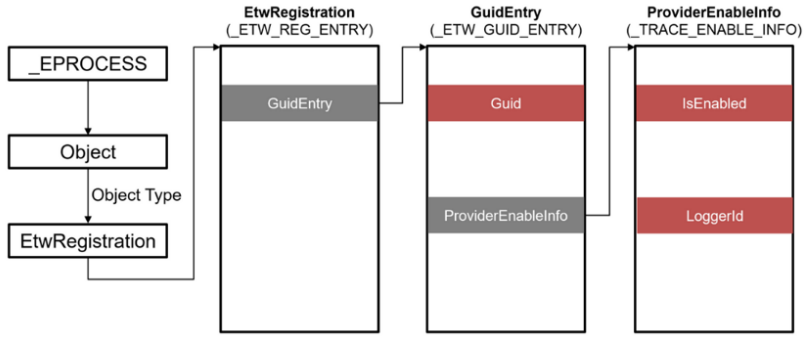

Image 2 Structure of ETW providers

Casey [2] discussed a forensic methodology where log analysis plays a central role in identifying intrusion patterns. The structured data from Event Viewer was found to be instrumental in linking user actions with system states.

Mitropoulos et al. [3] introduced real-time frameworks for log correlation and pattern recognition, which showed enhanced response times in forensic case studies. Their works highlight the importance of integrated structured logs with SIEM systems.

For the Microsoft Event Viewer documentation [4], all the logs are encoded in the EVTX binary format and that can be carries metadata which is very crucial for auditing and post-incident review.

SANS Institute [5] guides for the importance of Windows Security logs for object (Q2) in identifying credential-based attacks, brute-force attempts, and privilege escalations.

Log correlation and XML-based structuring were further examined by Carrier [6], who emphasized that logs must be preserved in a tamper-evident and verifiable format. XML schemas support this need.

Altheide and Carvey [7] offered detailed walkthroughs of interpreting logs within forensic suites such as FTK and Autopsy, supporting the need for consistent log formatting.

Reith et al. [8] and NIST [9] proposed frameworks and best practices for incorporating logs into forensic timelines.

Recent vendor whitepapers from CrowdStrike [10] and IBM X-Force [11] provided use cases where event logs aided in detecting advanced persistent threats (APTs).

Table 1 Discussion of Various Structure for Event Based Logs

Author	Year	Contribution	Relevance
Garfinkel, S.	2010	XML schema for forensic data	Log structuring for long-term analysis
Casey, E.	2011	Methodologies for forensic investigations	Log timeline construction
Mitropoulos, S.	2019	Real-time log correlation	Improved detection and SIEM integration
Microsoft Docs	2020	EVTX structure documentation	Metadata relevance for event parsing
SANS Institute	2022	Incident response playbooks	Log-based detection of brute-force attacks
Carrier, B.	2005	File systems and forensic principles	Secure and verifiable log storage
Altheide & Carvey	2011	Practical forensic log analysis	Usage of open-source tools
Reith et al.	2002	Forensic process models	Integration of logs in process models
NIST SP 800-86	2006	Guide to forensic techniques	Best practices for evidence collection
CrowdStrike	2023	Threat hunting with event logs	Case studies for event-based detection
IBM X-Force	2023	Case study-based forensic response	Correlation of logs with threat intelligence

4. ARCHITECTURE AND DATA FLOW

The following diagram shows the process flow of event log acquisition, XML conversion, and forensic analysis:

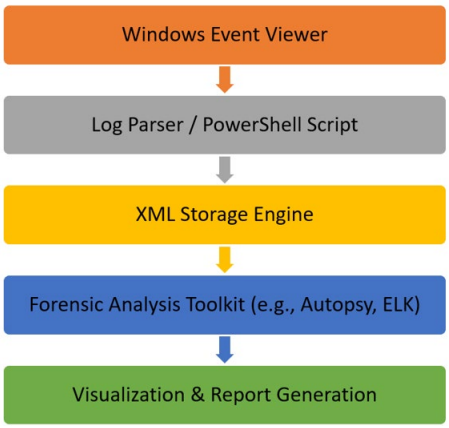

Image 3 Architecture for event viewer logs forensic Process

The Data Initially in row format collected using some scripts and those data can be parser to store in Specific XML format which can be used for the digital forensic purpose easy to analyze the record or logs of the event viewer.

5. IMPLEMENTATION

The implementation of XML-based event log analysis can be carried out using Windows PowerShell, EvtxECmd, or Python-based scripts. Below, we demonstrate a complete workflow using PowerShell and Python for extracting and processing Event Viewer data.

5.1. ENVIRONMENT SETUP

- Windows 10/11 (Administrator Access)

- PowerShell (Version 5 or above)
- Python 3.x (optional for extended parsing)
- Tools: Event Viewer, LogParser, EvtxECmd

5.2. STEP-BY-STEP LOG EXTRACTION AND XML CONVERSION (USING POWERSHELL)

Step 1: Extracting Logs by Category

```
# Application Logs (Q1)
Get-WinEvent -LogName Application | Export-Clixml -Path Q1_Application.xml
# Security Logs (Q2)
Get-WinEvent -LogName Security | Export-Clixml -Path Q2_Security.xml
# Setup Logs (Q3)
Get-WinEvent -LogName Setup | Export-Clixml -Path Q3_Setup.xml
# System Logs (Q4)
Get-WinEvent -LogName System | Export-Clixml -Path Q4_System.xml
# Forwarded Logs (Q5)
Get-WinEvent -LogName ForwardedEvents | Export-Clixml -Path Q5_Forwarded.xml
```

Step 2: Viewing XML Content

You can open the resulting XML files in Notepad++, XML Viewer, or Visual Studio Code. A snippet will look like:

```
<Event>
<EventID>4624</EventID>
<Level>Information</Level>
<TimeCreated>2023-07-25T14:23:05</TimeCreated>
<Source>Microsoft-Windows-Security-Auditing</Source>
<Message>An account was successfully logged on</Message>
</Event>
```

5.3. STEP-BY-STEP PARSING USING PYTHON

To analyze the XML files and filter specific patterns:

```
import xml.etree.ElementTree as ET
tree = ET.parse("Q2_Security.xml")
root = tree.getroot()
for event in root.findall(".//Event"):
event_id = event.find("EventID").text
level = event.find("Level").text
time_created = event.find("TimeCreated").text
message = event.find("Message").text
if event_id == "4625":
print(f"Failed Login - Time: {time_created}, Level: {level}, Message: {message}")
```

5.4. ADVANCED PARSING USING EVTXECMD

EvtxECmd is a powerful tool by Eric Zimmerman:

EvtxECmd.exe -d C:\Logs -o C:\ParsedLogs -f *.evtx --csv

The above command parses all EVTX files in a directory and exports CSV-format logs for timeline analysis.

5.5. TIMELINE RECONSTRUCTION

Once all logs are parsed, they can be sorted by TimeCreated and visualized:

- Use Excel or Pandas (Python) to create a timeline of events.
- Map failed logins (4625), successful logins (4624), shutdowns (1074), and application errors (1000).

```
import pandas as pd
import matplotlib.pyplot as plt
df = pd.read_csv("SecurityLog.csv")
df["TimeCreated"] = pd.to_datetime(df["TimeCreated"])
filtered = df[df["EventID"].isin([4625, 4624, 1074])]
filtered.groupby("EventID")["TimeCreated"].count().plot(kind='bar')
plt.title("Event Counts by Type")
plt.show()
```

5.6. STORAGE AND INTEGRITY

XML allows use of schema validation (XSD) to ensure integrity:

- Use XML Digital Signature for authenticity.

Store logs in WORM (Write Once Read Many) devices for forensic admissibility.

6. COMPARATIVE ANALYSIS WITH EXISTING STRUCTURES

To assess the efficiency, compatibility, and forensic soundness of the proposed XML-based log structure, we have compared it with three commonly used log storage formats:

Which are JSON, relational databases (SQL), and raw EVTX files with proposed structure

6.1. FEATURE-BASED COMPARISON

Table 2 Feature-Based Comparison

Feature	XML-Based Logs	JSON Logs	SQL Database	Raw EVTX Files
Human Readability	High	Medium	Low	Very Low
Schema Validation	Yes (via XSD)	No	Yes	No
Forensic Compatibility	High (used in tools)	Medium	High	Low
Tamper Detection	Medium (signable)	Medium	High	Low
Query Support	XPath	Custom/manual	SQL	None
Compression Efficiency	Medium	High	High	Low
Tool Integration	High (Autopsy, X-Ways)	Medium (SIEM tools)	High (Splunk, ELK Stack)	Very Low
Event Correlation	Strong	Average	Strong	Weak
Timeline Analysis Ready	Yes	Yes	Yes	No

6.2. OBSERVATIONS

- XML provides a balanced format supporting structure, validation, and ease of integration into forensic workflows.
- JSON is lightweight and easier to parse but lacks schema enforcement.
- SQL excels in query speed but needs complex setup and is less portable.
- Raw EVTX files are ideal for original log storage but are not suitable for immediate forensic processing.

7. CONCLUSION

Event Viewer is a cornerstone tool for Windows-based digital forensic investigations. By categorizing logs (Q1–Q5) and storing them in XML format, investigators can enhance visibility, correlation, and evidentiary value. Compared to unstructured or flat formats, XML provides flexibility, machine-readability, and compatibility with modern forensic tools. This paper demonstrated a practical and structured approach to using Event Viewer data for ensuring system integrity and detecting malicious behavior.

CONFLICT OF INTERESTS

None.

ACKNOWLEDGMENTS

None.

REFERENCES

Garfinkel, S. (2010). Digital forensics XML and structured storage. Digital Investigation.
Casey, E. (2011). Digital evidence and computer crime. Academic Press.
Mitropoulos, S., Karakoidas, V., Spinellis, D., & Louridas, P. (2019). Real-time event log analysis. IEEE Access.
Microsoft Docs. (2020). Event Viewer documentation. https://learn.microsoft.com
SANS Institute. (2022). Event log analysis. https://www.sans.org/white-papers/event-log-analysis/
Carrier, B. (2005). File system forensic analysis. Addison-Wesley.
Altheide, C., & Carvey, H. (2011). Digital forensics with open source tools. Syngress.
Reith, M., Carr, C., & Gunsch, G. (2002). An examination of digital forensic models. International Journal of Digital Evidence, 1(3).
National Institute of Standards and Technology. (2006). Guide to integrating forensic techniques into incident response (SP 800-86). https://csrc.nist.gov/publications/detail/sp/800-86/final
CrowdStrike. (2023). Endpoint detection and log management. https://www.crowdstrike.com
IBM X-Force. Event log analysis case studies. https://www.ibm.com/security/xforce
Mandia, K., Prosise, C., & Pepe, M. (2003). Incident response & computer forensics. McGraw-Hill.
Microsoft. (2023). LogParser tool documentation. https://learn.microsoft.com/en-us/sql/tools/logparser
Zimmerman, E.. EvtxECmd documentation. https://ericzimmerman.github.io
National Cyber Security Centre (UK). (2023). Windows event logging guidance. https://www.ncsc.gov.uk
Stallings, W. (2019). Computer security: Principles and practice. Pearson.
Kaspersky Labs. (2023). Best practices for log analysis. https://www.kaspersky.com
Sophos. (2022). Investigating Windows logs during threat hunts. https://www.sophos.com
AlienVault Labs. (2023). Log correlation techniques. https://cybersecurity.att.com
Patel, P. C. (2013). Aggregation of digital forensics evidences. Int J Comput Trends Technol (IJCTT), 4(4), 881-884.