Journal of COMPUTER AND FORENSIC SCIENCES

CFS 2025, Vol. 4, Issue 1, pp. 32-61
https://doi.org/10.5937/jcfs4-59602
Original research paper

Application of Time Series Algorithms in Cybersecurity

Marko M. Živanović[1*], Marjan D. Milošević[2], and Emilija Kisić[3]

[1] University Metropolitan, Faculty of Information Technology, Belgrade, Serbia, marko.zivanovic@metropolitan.ac.rs, 0009-0005-9264-3314

[2]University of Kragujevac, Faculty of Technical Sciences, Čačak, Serbia, marjan.milosevic@ftn.kg.ac.rs, 0000-0003-4730-1292

[3]University Metropolitan, Faculty of Information Technology, Belgrade, Serbia,emilija.kisic@metropolitan.ac.rs, 0000-0003-3059-2353

* Corresponding author: marko.zivanovic@metropolitan.ac.rs

Received: June 19, 2025 • Accepted: July 23, 2025 • Published: September 18, 2025

Abstract: This paper explores the application of time series algorithms to enhance anomaly detection in cybersecurity. Windows log files such as PowerShell Operational, Windows Defender, Firewall, System, and others were analyzed, focusing on those with the highest informational potential and data volume. Various models were used: exponential smoothing (Holt-Winters), Prophet, Fourier analysis, and Kalman filter for modeling seasonal, periodic, and linear patterns in system events. Advanced methods include LSTM and GRU neural networks, as well as ensemble algorithms like Random Forest and XGBoost, which demonstrated high accuracy in detecting unusual behavior. Special emphasis was placed on dynamic models, such as Bayesian Structural Time Series, to understand system states over time. Experiments show that applying multiple models enables a robust and adaptive approach to log analysis, especially for early detection of attacks and deviations from norms. The proposed framework highlights the importance of predictive analytics in preventive cybersecurity and provides a foundation for developing intelligent systems for real-time monitoring and response.

Keywords: time series, cybersecurity, LSTM, anomaly detection, XGBoost.

1. INTRODUCTION

In today's rapidly evolving digital landscape, cybersecurity has become a critical concern for organizations and individuals alike. With the increasing complexity and volume of cyber threats, traditional security measures are often insufficient for early detection and prevention of attacks. One promising approach to enhance cybersecurity is the use of time series analysis techniques applied to system log data. System logs, such as those generated by Windows operating systems—including PowerShell Operational, Windows Defender, Firewall, and System logs—contain rich temporal information that can be leveraged to identify abnormal patterns indicative of malicious activities or system faults. Time series algorithms offer powerful tools for modeling and predicting system behavior over time.

Citation: M. Živanović, M. Milošević, and E. Kisić, "Comparative Analysis of Machine Learning Models for Real-time Object Detection," *Journal of Computer and Forensic Sciences*, vol. 4, no. 1, pp. , pp. 32-61 2025, https://doi.org/10.5937/jcfs4-59602.

By capturing seasonal trends, periodic fluctuations, and unexpected anomalies in log data, these methods enable proactive threat detection and improved situational awareness. This paper explores a diverse set of time series techniques ranging from classical statistical models such as exponential smoothing (Holt-Winters), Fourier analysis, and Kalman filters to advanced machine learning approaches including Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) neural networks [1, 2, 3]. Additionally, ensemble learning methods like Random Forest and XGBoost are incorporated due to their proven effectiveness in handling non-linear and complex temporal dependencies [4]. Beyond static models, dynamic frameworks such as Bayesian Structural Time Series [5] and Markov Chains [6] are examined to capture evolving system states and temporal dependencies more effectively. The integration of these multiple modeling approaches aims to provide a robust and adaptive framework for analyzing log data [7], facilitating early detection of cyber attacks and deviations from normal system behavior. The significance of this research lies in its potential to advance predictive analytics in cybersecurity, enabling the development of intelligent monitoring systems that can operate in real-time, reduce false alarms, and enhance response strategies. By harnessing the temporal structure inherent in system logs, this study contributes to building more resilient defenses against increasingly sophisticated cyber threats. The remainder of this paper is organized as follows: *Materials and Methods* describes the data collection process, log sources, and statistical techniques used for analysis; *Results* presents the findings from applying time series models to Windows event logs; *Discussion* evaluates model performance and implications for cybersecurity; and *Conclusion* summarizes key insights and suggests future research directions.

2. MATERIALS AND METHODS

The log files used in this analysis were collected from a local Windows 10 Pro machine using a custom PowerShell script that leverages the **Get-WinEvent** cmdlet. The script was scheduled to run at fixed intervals, every 15 minutes, using the Task Scheduler. Data acquisition was conducted over a continuous period of 60 days, during which the system was used for typical daily activities, including web browsing, software development, system updates, and administrative tasks. This process allowed for the capture of both normal operating conditions and occasional anomalies. The resulting dataset encompasses 12 distinct log sources, with a combined total of 12,046 individual log entries exported in **.csv** format for further preprocessing and time series transformation.

The selected logs span a broad spectrum of system, security, and network events, providing a holistic view of the machine's activity. The general **Application** and **System** logs, each contributing 1,000 informational events, served as a baseline for standard operations and system health. For security-specific insights, several critical logs were included. The **Microsoft-Windows-PowerShell/Operational** log was the most voluminous with 4,007 entries, capturing detailed script execution activities and primarily consisting of "Warning" level events (e.g., Event ID 4104), making it a key source for analyzing command-line behavior. Network activity was monitored through the **Microsoft-Windows-Firewall**

with **Advanced Security/Firewall log,** which recorded 988 "Information" events related to enforced traffic rules, predominantly Event ID 2097 indicating allowed connections.

Further granularity was achieved by incorporating logs that track specific system functions. The **Microsoft-Windows-Windows Defender/Operational** log provided 1,000 informational events detailing routine antivirus scans and updates (most commonly Event ID 1151). User session activities were tracked via the **Microsoft-Windows-Winlogon/Operational** log (1,000 events), while the **Microsoft-Windows-NetworkProfile/Operational** log (1,000 events) recorded changes in network connectivity status (e.g., Event ID 4004). Logs with fewer but highly specific events, such as **Microsoft-Windows-DeviceSetupManager/Operational** (102 events) and **Microsoft-Windows-User Device Registration/Admin** (104 events), offered context on hardware changes and device enrollment. The low event count in the **Microsoft-Windows-RemoteDesktopServices-RdpCoreTS/Operational** log (16 events) suggested that remote access was infrequent during the observation period.

The number of recent events exported from each log file for analysis is detailed in Table 1. For each exported event, the following attributes were included: **TimeCreated** (the timestamp of the event), **Id** (the event identifier), **LevelDisplayName** (the severity level, e.g., "Information" or "Warning"), and **Message** (the event description, when available). The complete dataset used in this study is not publicly released due to privacy and security considerations. However, it is available from the corresponding author upon reasonable request for research purposes, under appropriate ethical guidelines.

Table 1. *Log sources used in predictive modeling of Windows events.*

Log Name	Description	Significance for Analysis
Security	Records security-related events such as login attempts, privilege changes, and resource access.	Crucial for detecting unauthorized access attempts and monitoring security incidents.
System	Contains information about system components and drivers, including errors and warnings.	Helps identify issues with hardware and core system services.
Application	Logs events generated by applications installed on the system.	Useful for tracking application performance and errors.
Windows Firewall with Advanced Security	Tracks firewall activity, including blocked and allowed connections.	Important for analyzing network security and identifying potential threats.
DNS Client Operational	Logs DNS client activity, such as queries and responses.	Helps diagnose name resolution and network connection issues.
DHCP Client Operational	Monitors DHCP client activity, including IP address assignments.	Useful for resolving network configuration and connectivity issues.
Sysmon Operational	Records detailed system activity, including process creation and network connections.	Enables advanced threat detection and forensic analysis.
PowerShell Operational	Logs PowerShell usage, including script and command execution.	Important for detecting potentially malicious scripts and automated attacks.
Windows Defender Operational	Tracks Windows Defender activity, including scans and threat detections.	Essential for monitoring antivirus protection and identifying malware.

Terminal Services Local Session Manager	Logs events related to Remote Desktop sessions.	Useful for monitoring access to remote sessions and unauthorized attempts.
Windows Update Client Operational	Tracks activity of the Windows Update client, including installations and errors.	Helps identify issues with system updates.
Task Scheduler Operational	Records scheduled task executions and their statuses.	Important for tracking automated processes and potentially malicious tasks.
Kernel-Power	Logs system power events, such as unexpected shutdowns.	Useful for diagnosing power and system stability issues.
Winlogon Operational	Tracks user login activity on the system.	Helps monitor user sessions and potential security incidents.
Application Experience – Program Inventory	Logs information about installed applications and their changes.	Useful for software inventory and detecting unauthorized installations.
Certificate Services Client Lifecycle System	Monitors certificate-related activity and renewals.	Important for managing certificate-based security infrastructure.
User Device Registration Admin	Logs user device registration events.	Helps manage devices and user access.
User Device Registration Operational	Tracks operational aspects of user device registration.	Useful for diagnosing device registration issues.
Network Profile Operational	Logs changes in network profiles and connections.	Helps monitor network changes and potential security risks.
Network Connection Operational	Tracks establishment and termination of network connections.	Important for analyzing traffic and detecting unauthorized access.
Network Isolation Operational	Logs network traffic isolation events.	Useful for analyzing network segmentation and security.
Network Sharing Operational	Tracks network resource sharing activity.	Helps identify potentially insecure resource sharing.
Group Policy Operational	Logs application and changes of group policies.	Crucial for managing system configuration and security policies.
Windows Defender Antivirus Operational	Tracks activity of the Windows Defender antivirus engine.	Important for malware detection and response.
Windows Defender ATP Operational	Logs advanced threats detected by ATP.	Enables detection of sophisticated attacks and threats.
User Profiles Service Operational	Monitors loading and management of user profiles.	Helps resolve issues with user profiles and sessions.
Device Setup Manager Operational	Logs device installation and configuration events.	Useful for diagnosing hardware and driver-related issues.
Security-SPP Operational	Monitors software license protection activity.	Important for license management and software protection.
Diagnostics Performance Operational	Logs system performance and potential bottlenecks.	Helps optimize performance and identify issues.
Remote Access Operational	Tracks remote access activity.	Crucial for monitoring VPN connections and remote logins.
Remote Desktop Services – RdpCoreTS Operational	Logs Remote Desktop Protocol (RDP) session activity.	Important for detecting and tracking remote sessions.
Remote Desktop Services – RdpCoreTS Debug	Provides detailed debug information for RDP sessions.	Useful for in-depth troubleshooting of remote desktop issues.

In the scope of this research, special attention is given to a subset of event logs deemed particularly relevant for system security, operational integrity, and anomaly detection. The

selected logs provide a multifaceted view of system activity, user behavior, and security-related events. Below is an overview of the key logs included in the analysis:

• **System –** Contains events related to system-level operations, including driver errors, service startups, and hardware-related notifications. This log is essential for tracking the health and stability of the operating system over time.

• **Application –** Captures events generated by user-installed applications. Analyzing this log helps in identifying software crashes, misconfigurations, or abnormal usage patterns.

• **Security –** One of the most critical logs for forensic and auditing purposes, it includes login attempts, privilege use, and access control events. It is fundamental for tracking unauthorized access attempts and auditing user activity.

• **Microsoft-Windows-PowerShell/Operational** – Provides a record of PowerShell script executions, including commands issued interactively or via scripts. This log is especially important for detecting advanced persistent threats (APTs) and automation misuse, as PowerShell is a common vector in cyberattacks.

• **Microsoft-Windows-Windows Defender/Operational –** Logs antivirus scans, malware detections, and remediation actions taken by Windows Defender. This log is crucial for identifying security threats and understanding how the system reacts to them.

• **Microsoft-Windows-Windows Firewall with Advanced Security/Firewall –** Captures data on allowed and blocked network traffic as determined by firewall rules. It provides insights into external connections and can reveal attempts to breach network defenses.

• **Microsoft-Windows-NetworkProfile/Operational** – Records changes in network connectivity and profile assignments (e.g., private, public). This log is helpful for tracking when and how a machine switches between networks—often an indicator of mobile use or exposure to unknown networks.

• **Microsoft-Windows-DeviceSetupManager/Operational –** Contains entries related to hardware detection and driver installation. This can be used to identify unauthorized device usage or to correlate issues caused by newly introduced hardware.

• **Microsoft-Windows-RemoteDesktopServices-RdpCoreTS/Operational –** Logs activities related to Remote Desktop Protocol (RDP) sessions. It plays a key role in monitoring remote access, which is frequently exploited in targeted attacks and lateral movement.

• **Microsoft-Windows-WindowsUpdateClient/Operational –** Records system update events, including successful or failed updates. This information is useful for ensuring system compliance and can also highlight moments of increased system load or instability.

Together, these logs offer a comprehensive temporal dataset suitable for anomaly detection, time-series modeling, and correlation analysis across security, application, and system layers. Their diversity ensures a holistic view of the environment, enabling deeper insights into both routine operations and irregular behaviors.

For the purposes of this analysis, fundamental descriptive and inferential statistical models were applied to quantify the characteristics of event time series extracted from operating system log files. All logs were analyzed using statistical metrics, including **count**, **mean**, **median**, **standard deviation**, **variance**, **range** (min–max), **quartiles** (Q1, Q3), **interquartile range** (IQR), **coefficient of variation** (CV), **skewness**, **kurtosis**, **first-order autocorrelation**, **linear hourly trend**, and the **most frequent event type** and **ID**.

The arithmetic mean represents the central value of a dataset and is used to describe the average number of events within the observed time intervals.

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \quad , \tag{1}$$

Where x_i is the i-th value in the observed dataset, n is the total number of observations, $\bar{x}$ is the arithmetic mean [10].

Standard deviation quantifies the average deviation of values from the mean and is one of the most important measures of data dispersion.

$$\sigma = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2} \quad , \tag{2}$$

Where $(x_i - \bar{x})^2$ is the squared deviation of each value from the mean, $n-1$ is Bessel's correction for estimating population variance from a sample [11];

Variance is the square of the standard deviation and serves as a measure of total dispersion within the dataset. Variance is particularly useful when comparing variability across different logs, especially in analytical modeling contexts [12].

$$Var(X) = \sigma^2 \tag{3}$$

The coefficient of variation is a relative measure of dispersion and is used to compare variability across datasets with different means. CV is a dimensionless quantity, usually expressed as a percentage. A value of CV > 1 indicates high variability, while CV < 0.5 suggests a stable and predictable pattern [13].

$$CV = \frac{\sigma}{\bar{x}} \tag{4}$$

Skewness measures the degree to which a dataset deviates from a symmetric normal distribution. Skewness is particularly useful for detecting extreme values and anomalies in log data [14].

$$Skew(X) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{x_i - \bar{x}}{\sigma}\right)^3 \quad , \tag{5}$$

Interpretation:

$skew > 0$ — the distribution is right-skewed;

$skew < 0$ — the distribution is left-skewed;

$skew \approx 0$ — approximately symmetric distribution.

Kurtosis quantifies how much of the data is concentrated near the mean compared to a normal distribution. It is used to detect "peakedness" and the presence of heavy tails [15].

$$Kurt(X) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{x_i - \bar{x}}{\sigma}\right)^4, \tag{6}$$

Interpretation:

$kurt > 3$ — leptokurtic distribution (sharp peak, heavy tails);

$kurt < 3$ — platykurtic distribution (flat distribution);

$kurt \approx 3$ — normal distribution.

High kurtosis may indicate rare but intense events in the logs (e.g., bursts of PowerShell executions or spikes in network access).

Autocorrelation measures the correlation between a value and its preceding value in a time series, revealing patterns or periodicity in the data [16].

$$\rho_1 = \frac{\sum_{t=2}^{n}(x_t - \bar{x})(x_{t-1} - \bar{x})}{\sum_{t=2}^{n}(x_t - \bar{x})^2}, \tag{7}$$

Interpretation:

$\rho_1 > 0$ — positive autocorrelation;

$\rho_1 < 0$ — negative autocorrelation;

$\rho_1 \approx 0$ — no temporal dependency detected.

In system logs, autocorrelation can help identify recurring cycles (e.g., scheduled tasks or scanning behavior). To detect trends in event counts over time, linear regression is applied to the number of events per unit of time (e.g., per hour) [17]. The basic regression model is:

$$y = \beta_0 + \beta_1 t + \varepsilon, \tag{8}$$

Where:

y — the number of events;

t — time (in hours);

β_0 — intercept (initial value);

β_1 — slope (rate of change);

ε — error term.

A positive β1 indicates an increasing trend in activity over time, while a negative value suggests a decline.

Quartiles divide a dataset into four equal parts:

Q_1 — first quartile (25th percentile);

Q_2 — median (50th percentile);

Q_3 — third quartile (75th percentile).

The interquartile range (IQR) is defined as:

$$IQR = Q3 - Q1, \tag{9}$$

IQR measures the spread of the middle 50% of the data and is commonly used in outlier detection. Values outside the range $[Q_1 - 1.5 * IQR, Q_3 + 1.5 * IQR]$ are considered potential outliers [18].

3. STATISTICAL ANALYSIS OF KEY EVENT METRICS

The analysis of four key event metrics across the log files reveals important patterns in system behavior. First, the standard deviation graph (**Figure 1**) shows how much the number of events varies per hour: PowerShell and Firewall logs exhibit very high variability (94.57 and 70.88), indicating sudden and irregular spikes in activity, whereas Windows Defender shows stability with the lowest deviation (~2.4). Next, the analysis of hourly event trends (**Figure 2**) reveals that PowerShell records a strong negative trend (−12.82), suggesting a decline in activity following an initial "burst" (e.g., script executions), while logs such as System and Application remain mostly stable, without a pronounced trend. The coefficient of variation (std/mean) (**Figure 3**) further confirms instability in Firewall (1.79), PowerShell (1.66), and Application (1.52), while Defender is the most stable (~0.78), indicating consistent behavior over time. Finally, the lag-1 autocorrelation analysis (**Figure 4**) shows that the Winlogon log is strongly temporally correlated (0.426), likely due to cyclical user logins, whereas PowerShell shows a negative correlation (−0.327), a clear indicator of "burst" activity—if one hour was active, the next probably was not. **In conclusion**, these insights are of great importance for anomaly detection and predictive modeling applications, where logs with stable or predictable patterns (such as Winlogon or Defender) may be better suited for classical models like ARIMA/SARIMA. ARIMA (AutoRegressive Integrated Moving Average) and SARIMA (Seasonal ARIMA) are statistical models for time series analysis that combine autoregression, integration (differencing to achieve stationarity), and moving averages, with SARIMA additionally incorporating seasonal components. **On the other hand**, logs with high variability may require more robust or specialized approaches [19, 20, 21].

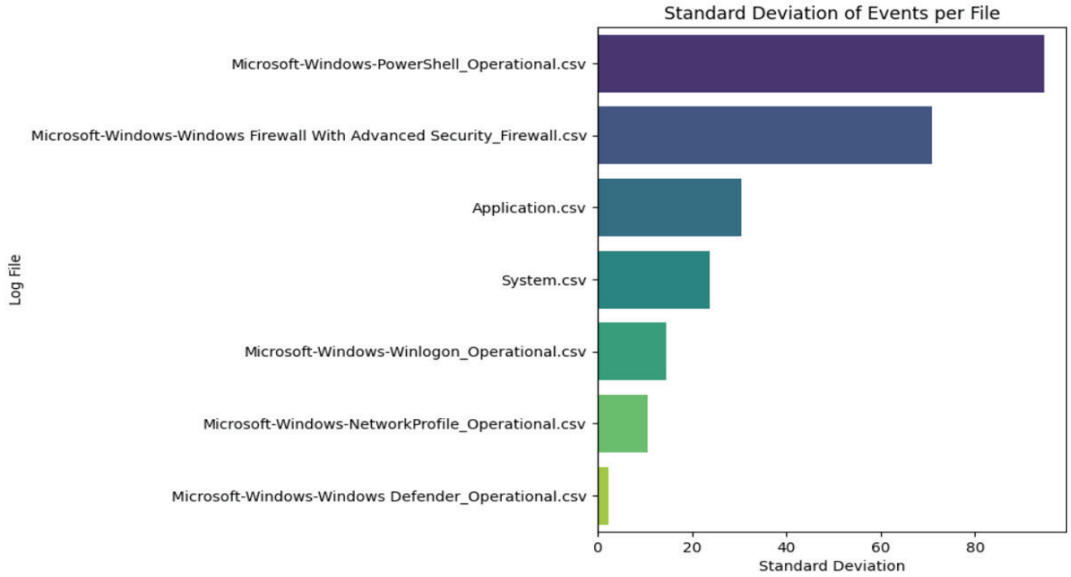

Figure 1. Standard Deviation of Events per File.

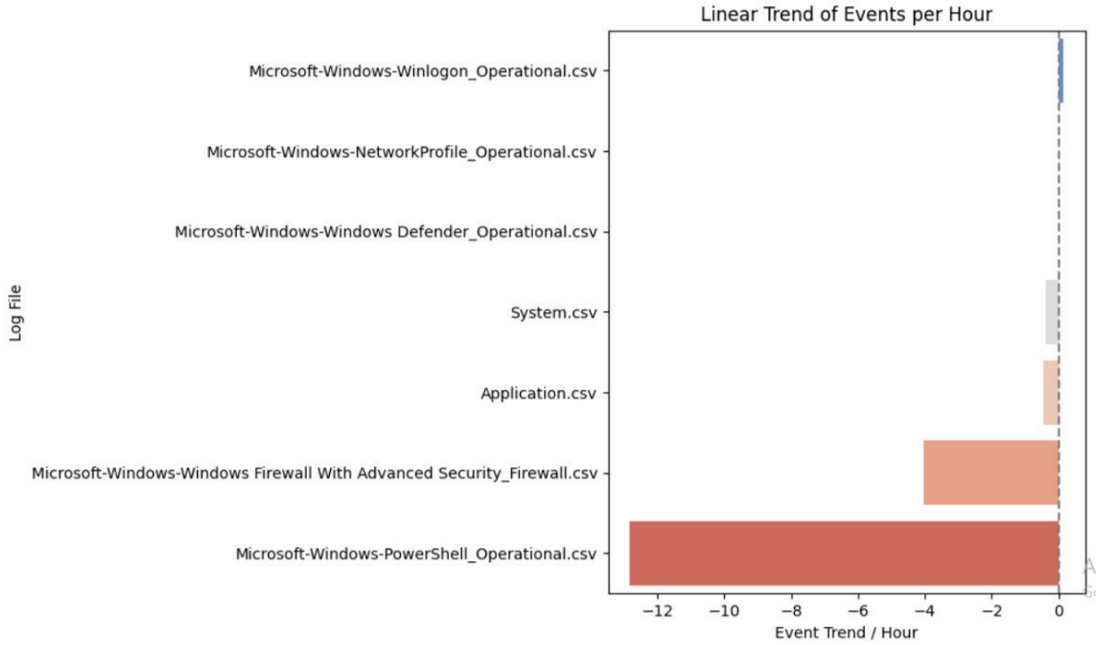

Figure 2. Linear Trends of Events per Hour.

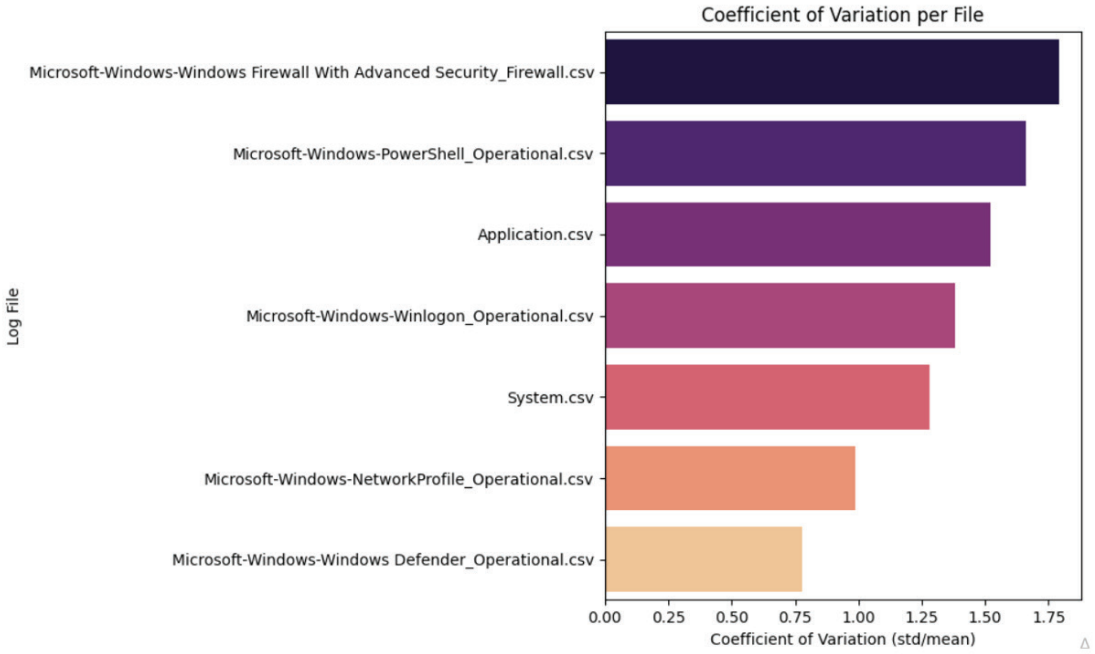

Figure 3. Coefficient of Variation per File.

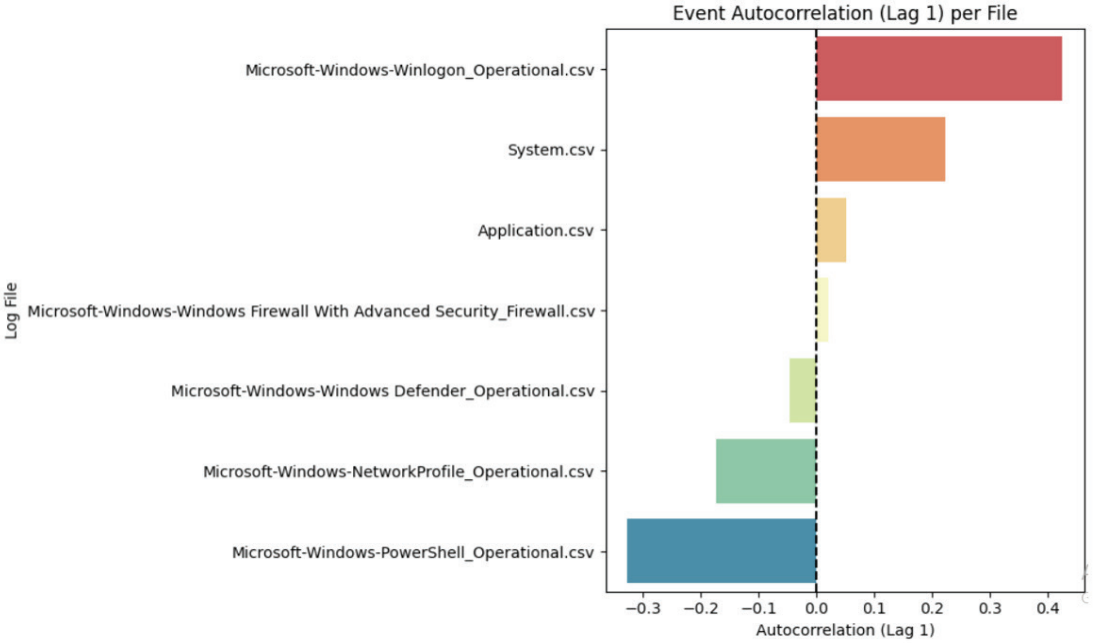

Figure 4. Event Autocorrelation per File.

3.1. Microsoft-Windows-PowerShell/Operational

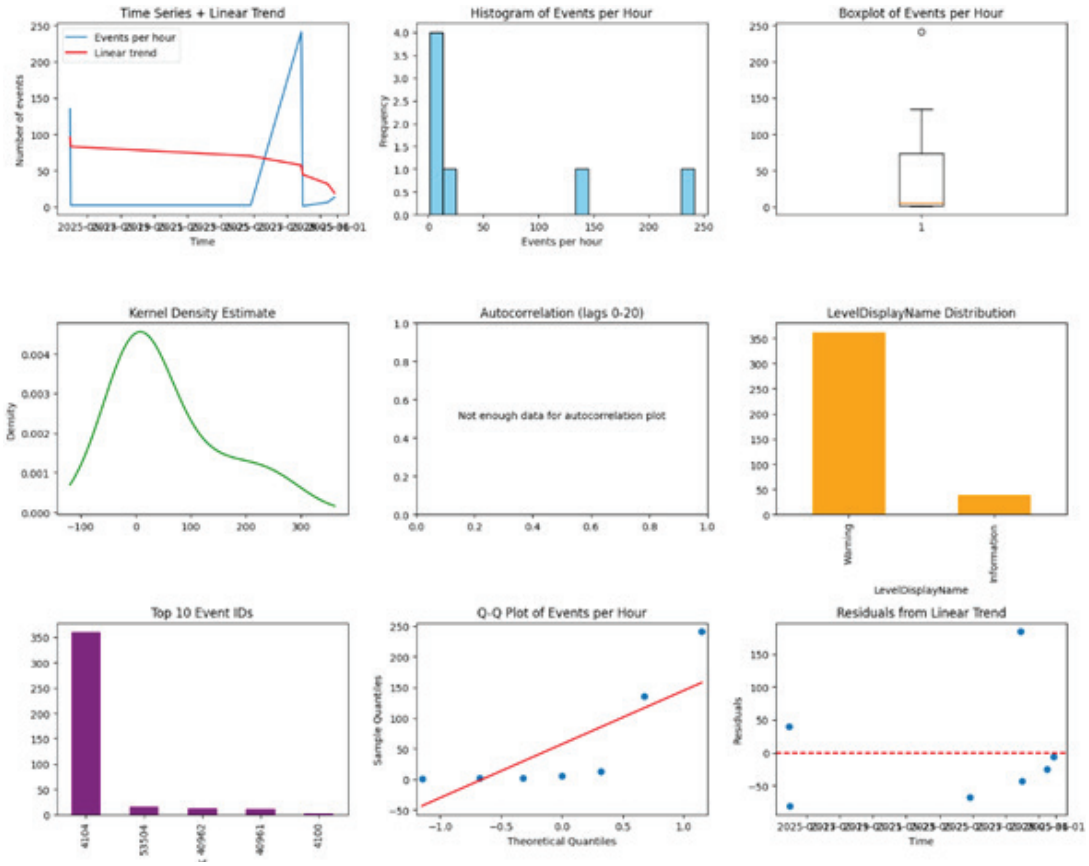

Figure 5. Microsoft-Windows-PowerShell/Operational Statistics.

As presented in Figure 5, analysis of the displayed charts reveals a clear pattern of PowerShell activity that is mostly nonexistent, except for one sudden and highly intense spike. The time series plot clearly illustrates this event through a long period of zero activity, which ends with a sharp jump to over 250 events in a single hour. Although the mathematically calculated trend line shows a decline, it is misleading, as it is heavily influenced by the long period of inactivity rather than the spike itself.

The unusual nature of this event is confirmed through multiple distribution charts; the histogram, boxplot, kernel density plot, and Q-Q plot unanimously indicate that the data drastically deviate from a normal, regular distribution, given that almost all activity is concentrated in a single extreme point (outlier). A key insight into the nature of this event is provided by the identification of the logs themselves: the "Top 10 Event IDs" chart reveals that nearly all activity originated from Event ID 4104, which specifically logs the execution of PowerShell scripts. Moreover, the majority of these events are classified as "Warning," indicating that the system itself flagged this activity as something to pay attention to. The conclusion, therefore, is that the charts do not depict regular activity but rather an isolated case of massive execution of an automated script.

3.2. Microsoft-Windows-Windows Defender/Operational

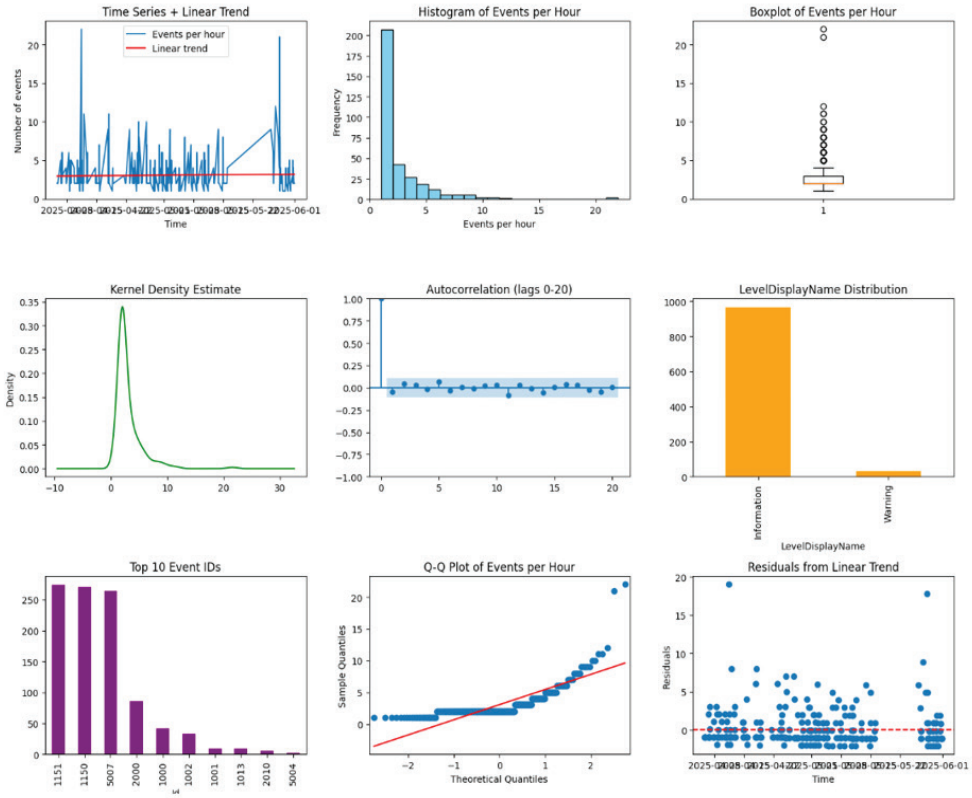

Figure 6. Microsoft-Windows-Windows Defender/Operational Statistics.

As presented in Figure 6, a **comprehensive analysis of the charts shows that the activity of the Defender antivirus program is constant and stable, but with frequent, occasional spikes.** The time series plot displays low but steady activity with regular spikes, while the flat red trend line confirms that there is no long-term increase or decrease in activity. This nature of "constant operation with occasional load" is further supported by the distribution charts; the histogram, boxplot, and kernel density plot clearly show that most events are of low frequency, but with a long "tail" of less frequent yet more intense activity, creating strong positive skewness.

The fact that the data do not follow an ideal "normal" distribution, as seen in the Q-Q plot, is entirely expected for this type of operation. Key context is provided by examining the type and level of events: the "Top 10 Event IDs" chart shows that the activity originates from various operations, with dominant Event IDs 1151 and 1150, which point to routine protection functions.

Most importantly, the "LevelDisplayName Distribution" chart reveals that the vast majority of these events are classified as mere "Information," rather than "Warning." Therefore, these charts do not indicate a problem but rather depict a portrait of a healthy and active antivirus solution performing its regular duties, such as scanning or updating, and generating informational logs in the process.

3.3 Microsoft Windows-Windows Firewall with Advanced Security/Firewall

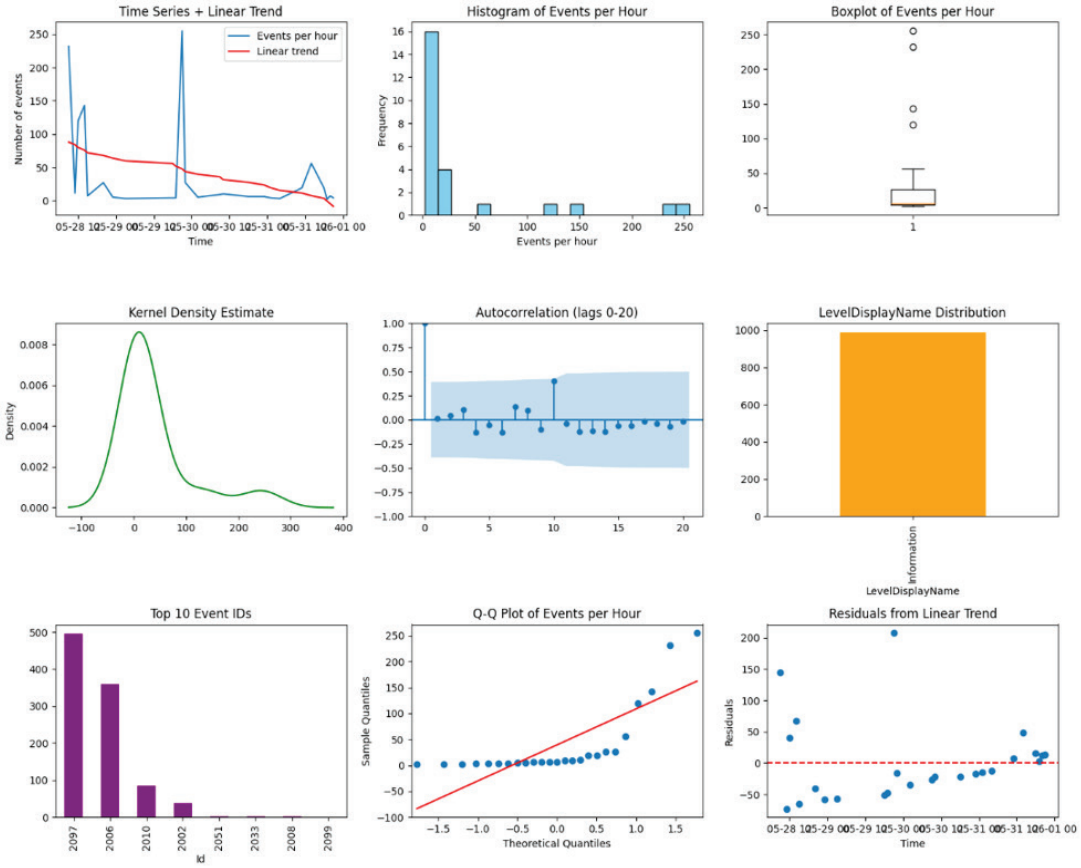

Figure 7. Microsoft-Windows-Windows Firewall with Advanced Security/Firewall Statistics.

The analysis of the Windows Firewall charts reveals activity marked by several pronounced but rare spikes in the number of events, which became less frequent over time. The time series plot clearly illustrates these spikes, as well as a significant downward trend (red line), suggesting that the overall frequency of events decreased during the observed period. The distribution of these events, as shown in the histogram, boxplot, and kernel density plots, confirms that most hours had very little activity, while a few hours experienced an extremely high number of events, resulting in strong positive skewness. However, a key piece of information that alters the interpretation of these "intense events" comes from analyzing their nature. The "LevelDisplayName Distribution" chart unambiguously shows that every single one of these events is classified as "Information," with not a single "Warning" or "Error" present. Furthermore, the "Top 10 Event IDs" chart reveals that the dominant events are ID 2097 and 2006, which typically refer to specific network operations or rule status updates, rather than blocked attacks. The conclusion is therefore more precise: although the firewall recorded occasional "storms" of events, they did not represent security threats. Instead, they were instances of mass generation of informational logs related to specific, likely automated, network processes whose frequency decreased over time.

3.4. Application

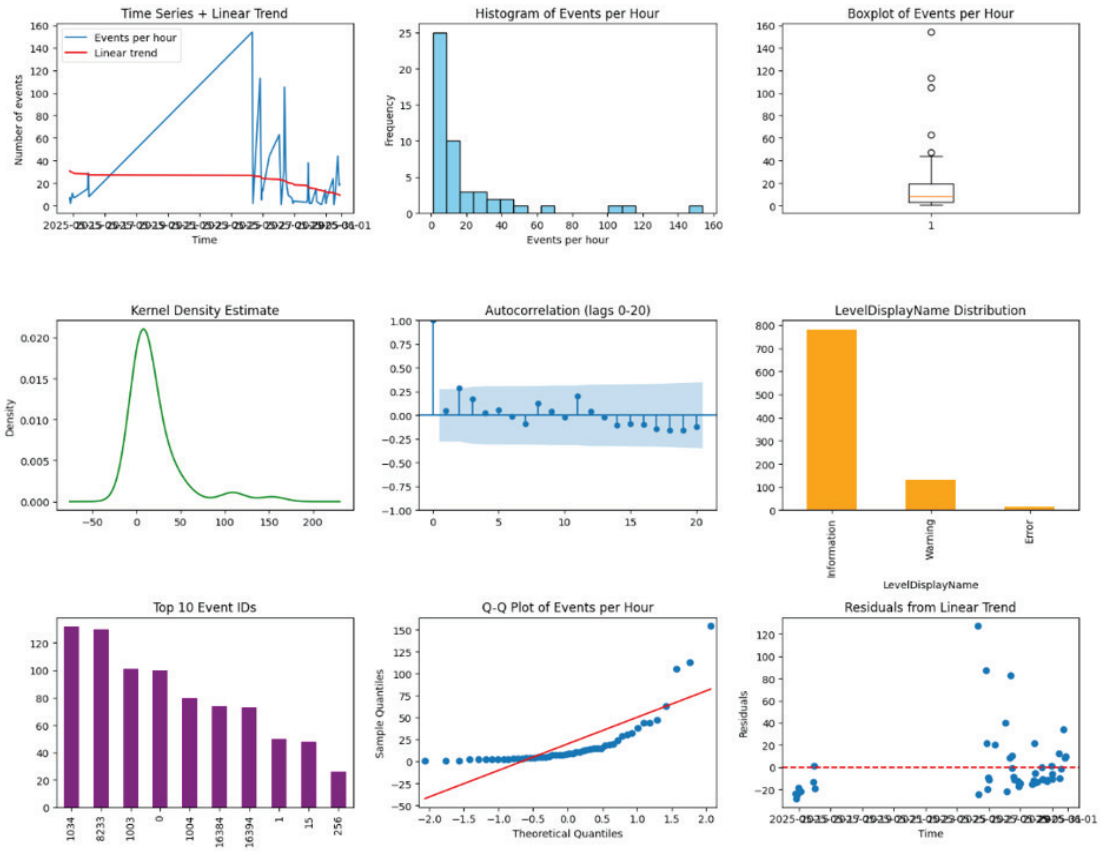

Figure 8. Application Statistics.

An analysis of the application messages and error charts reveals highly unstable activity characterized by occasional but intense spikes in the number of events. Statistically, the data shows a mean of approximately 20 events per hour but with a significant positive skewness of ~2.8 and high kurtosis of ~7.99, indicating that while most hours have low activity, there are frequent and extreme deviations. While the time series and distribution plots (Histogram, Boxplot) visualize this volatile, right-skewed pattern with numerous outliers, the key insight comes from the nature of the events themselves. The "LevelDisplayName Distribution" chart shows that the overwhelming majority of events are informational, not critical errors. However, the "Top 10 Event IDs" (such as 1034, 8233, and 1003) link these informational storms to specific application processes and instability, such as events preceding or following application crashes. Therefore, although the spikes are not composed of critical errors, their origin in software misbehavior makes this log category extremely valuable for diagnosing issues like crash dumps (Figure 8).

3.5. Microsoft Windows Network Profile/Operational

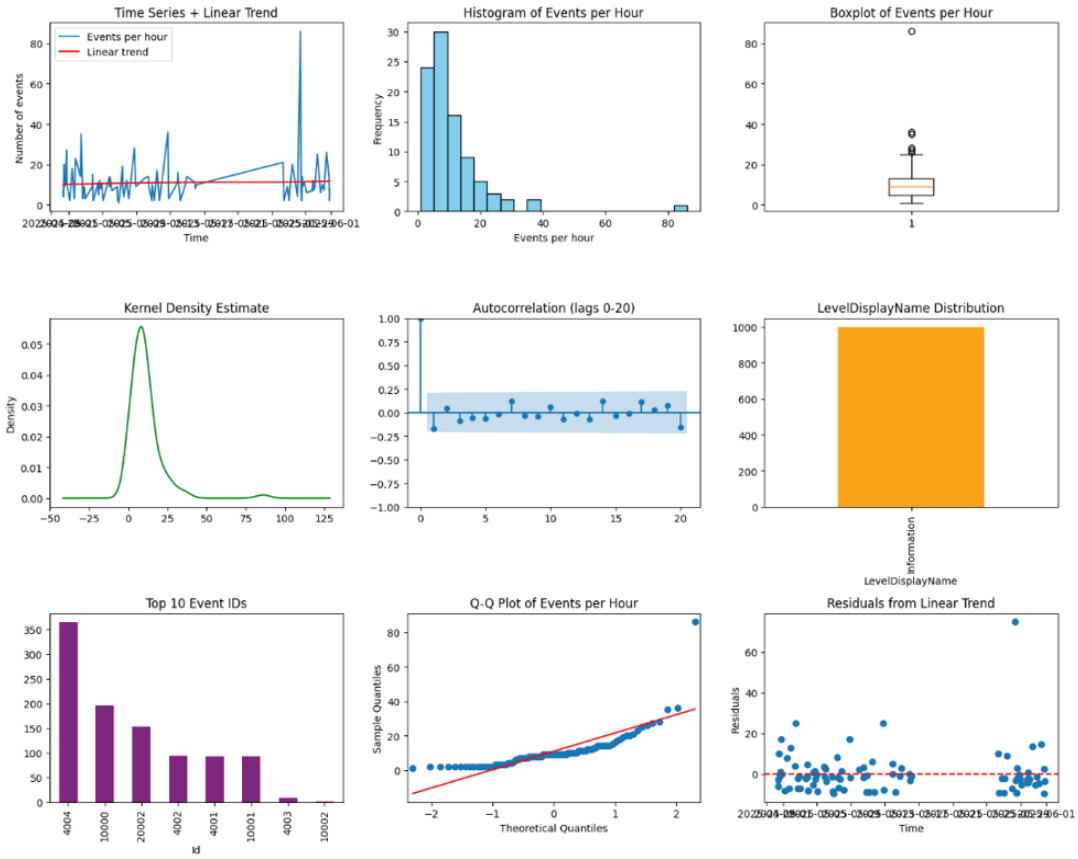

Figure 9. Microsoft-Windows-NetworkProfile/Operational Statistics.

The analysis of the provided charts indicates highly unstable activity, characterized by sporadic but very intense spikes in the number of events. The time series shows periods of low activity interrupted by sudden peaks reaching up to 80 events per hour. The linear trend is nearly flat, suggesting that there is no clear increase or decrease in average activity over the observed period—volatility is the dominant characteristic. The statistical distribution of events per hour, as displayed in the histogram and boxplot, clearly reveals strong right (positive) skewness. Most hours record a very low number of events (with a median below 10), but occasional extremes are evident through numerous outliers. The Q-Q plot further confirms that the data significantly deviate from a normal distribution, which is typical for datasets with rare but extreme events. A key insight is gained through the analysis of the nature of the events themselves. The "LevelDisplayName Distribution" chart unambiguously shows that all recorded events are informational in nature ("Information"), with no critical errors or warnings. Further analysis via the "Top 10 Event IDs" chart reveals that these informational spikes are predominantly driven by specific events, particularly Event ID 4004, which appears significantly more frequently than others. Additionally, the Autocorrelation Function (ACF) plot shows no statistically significant correlation between the number of events in consecutive hours. This suggests that the activity

spikes are random and unpredictable, rather than part of a recurring cycle. In conclusion, although the system does not report any critical errors, it generates occasional "information storms" (log storms) originating from very specific processes (e.g., the one related to ID 4004). While this behavior does not indicate system failure, it may be a symptom of inefficiencies or configuration issues. Monitoring these events is essential for optimizing performance and reducing log "noise," which can help in detecting real problems when they occur (Figure 9).

3.6. System

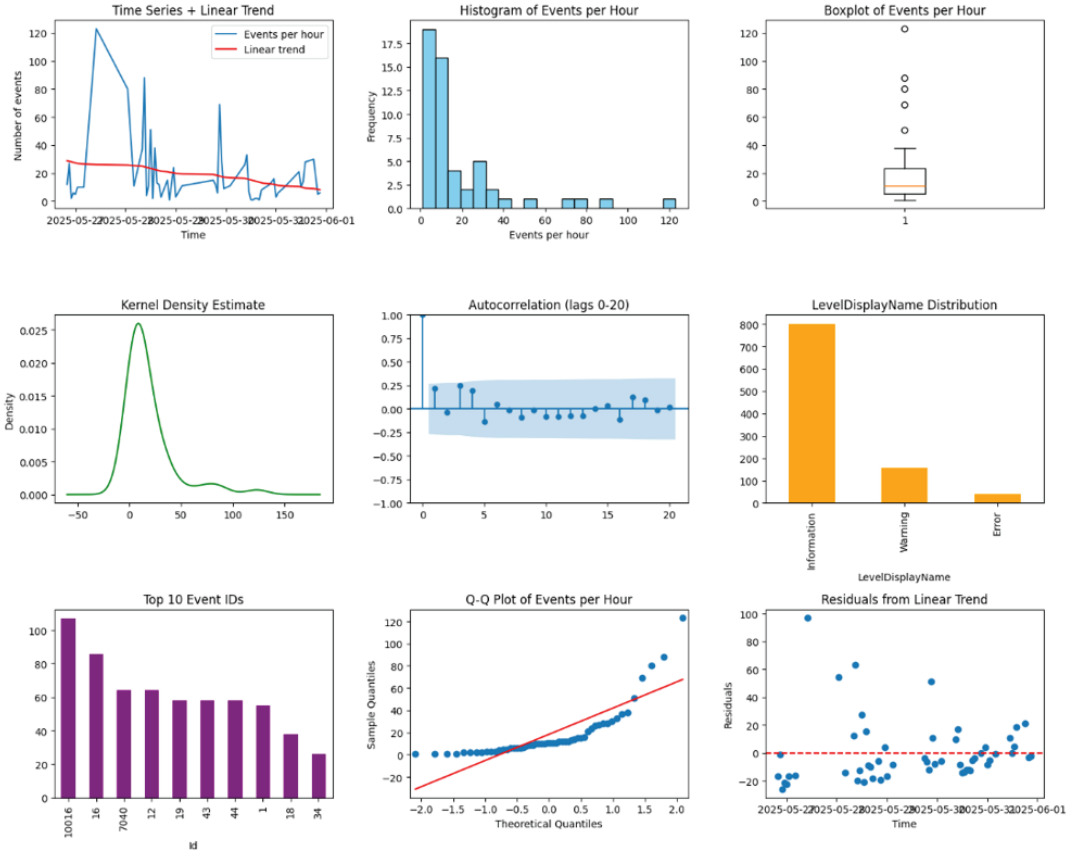

Figure 10. System Statistics.

The analysis of the presented charts reveals a highly variable nature of system events, with one prominent initial spike in activity exceeding 100 events per hour. Unlike the previous example, a clear downward linear trend is observed here, suggesting that the average number of events decreased over the observed period—likely as a result of some intervention or system stabilization. The distribution of events per hour, as shown in the histogram, boxplot, and KDE plot, exhibits pronounced right skewness. Most hours register a low number of events, while rare but intense spikes create a long right tail and numerous outliers. The Q-Q plot confirms that the data do not follow a normal distribution. The key difference

and most important insight lie in the nature of the events themselves. The "LevelDisplayName Distribution" chart shows that, although informational events ("Information") are the most frequent, the system also generates a significant number of warnings ("Warning") and, to a lesser extent, errors ("Error"). This points to actual issues within the application, rather than mere "noise" in the logs. The "Top 10 Event IDs" chart shows that the activity is generated by several different events, with Event ID 10016 being the most frequent. The Autocorrelation Function (ACF) chart, similar to the previous case, shows no temporal dependence, indicating that the occurrence of these events cannot be predicted based on prior activity. In conclusion, the data suggest the presence of instability or issues at the beginning of the observed period, which gradually subsided (as indicated by the downward trend). The presence of warnings and errors, although in the minority, makes these logs particularly important for diagnostics. Analyzing the events behind IDs such as 10016 and 7040—both associated with warnings—is key to understanding the root cause of the problem and preventing its recurrence (Figure 10).

3.7. Winlogon/Operational

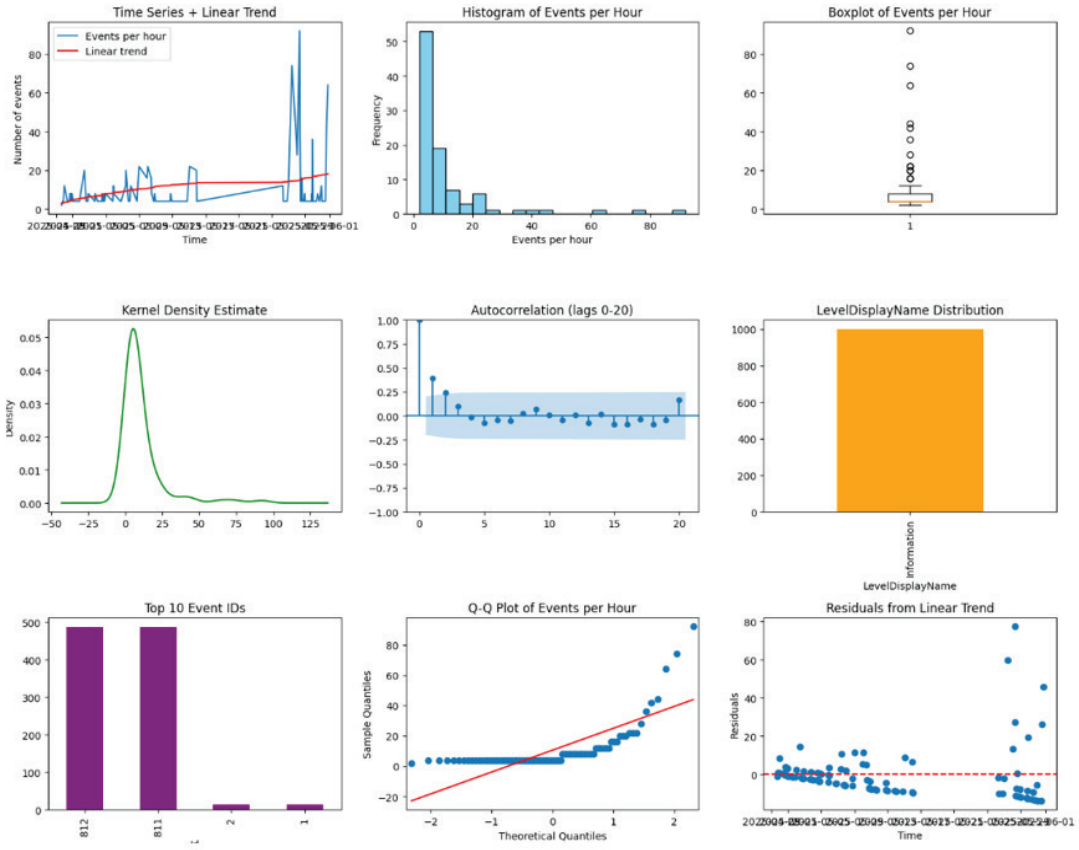

Figure 11. Winlogon/Operational Statistics.

As presented in Figure 11, t**he analysis of the provided charts reveals a pattern of activity characterized by extreme instability and occasional, very strong spikes in the number of events.**The time series shows that, in addition to random peaks exceeding 80 events, there is also a slight upward linear trend. This indicates a gradual increase in the average number of events over the analyzed period.

The statistical distribution is highly right-skewed, as seen in the histogram, boxplot, and KDE plot. The vast majority of hours record a minimal number of events (with a median close to zero), while extreme values occasionally occur, manifested as numerous outliers. The Q-Q plot definitively confirms that the data do not follow a normal distribution due to the heavy right tail. A key insight comes from the analysis of the type and source of events. The "LevelDisplayName Distribution" chart shows that 100% of the events are informational in nature ("Information"). Even more importantly, the "Top 10 Event IDs" chart reveals that nearly all activity is generated by just two IDs: 812 and 811. These two events account for the vast majority of all logs, meaning the "information storms" originate from a highly concentrated source. The Autocorrelation Function (ACF) plot suggests that the activity is mostly random. While there is a borderline significant correlation at the first time lag (lag 1), it is not strong enough to indicate a clear, predictable pattern. Thus, the activity spikes can still be considered unpredictable. In conclusion, the system does not report any errors but suffers from extreme "chattiness" caused by two specific processes (associated with IDs 812 and 811). The upward trend in this activity suggests that the problem may become more pronounced over time. Although not critical, such behavior unnecessarily burdens the logging system and complicates monitoring. Investigating the cause of these messages is recommended in order to reduce informational "noise" and optimize application performance.

3.8. Terminal Services Local Session Manager/Operational

This analysis covers a longer time period (approximately 10 months) and reveals generally low but stable activity. The time series shows that the number of events per hour is mostly in the single digits, with occasional minor spikes reaching a maximum of 35. The linear trend is completely flat, confirming that there is no long-term increase or decrease in activity; the system is in a stable, though occasionally "noisy," state. All distribution charts (Histogram, Boxplot, KDE) indicate extreme right skewness. The vast majority of data is clustered around zero, meaning that there are often hours with no events at all. Rare events appear as outliers. The Q-Q plot further confirms a significant deviation from a normal distribution. Key insights come from the analysis of the nature and temporal structure of the events: the **event type** shown in the "LevelDisplayName Distribution" chart indicates that the events are predominantly informational, but it is important to note that errors ("Error"), although very rare, are also recorded and require attention. The **source of the events**, according to the "Top 10 Event IDs" chart, shows that Event ID 40 is the most frequent trigger of activity, followed by several others with lower frequency. As for **temporal patterns**, unlike previous analyses, the Autocorrelation Function (ACF) chart here reveals statistically significant correlation at multiple time lags (e.g., lags 1, 16, 18), sug-

gesting that the events are not entirely random and that a subtle periodic process may be driving them. In conclusion, the system is stable over the long term and shows no signs of escalating problems. However, it is not without flaws. The presence of even rare errors calls for further investigation. The significant autocorrelation suggests the existence of hidden, periodic patterns in system behavior. It is recommended to examine the processes related to the most frequent event IDs (especially ID 40) and analyze the conditions under which errors and periodic events occur in order to proactively address them before they develop into more serious issues. (Figure 12).

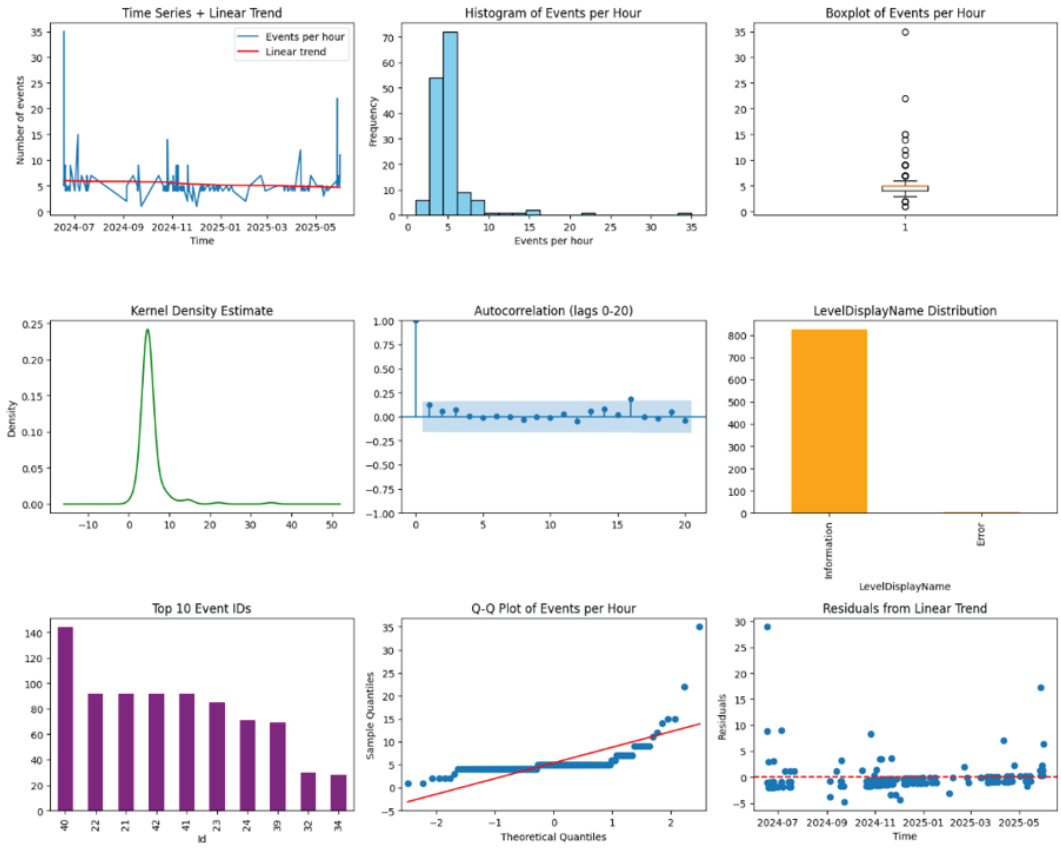

Figure 12. TerminalServices-LocalSessionManager/Operational Statistics.

3.9. Windows Update Client/Operational

The presented data illustrate an initial period of system instability, marked by a sharp spike in activity, followed by a clear trend of stabilization. The time series shows that, despite occasional fluctuations, the downward linear trend indicates a decrease in the average number of events during the observed period ending in late May 2025. Statistically, the data are highly right-skewed, as confirmed by the histogram, boxplot, and KDE; most activity is concentrated at very low levels, with rare but extreme outliers. A key finding emerges from the "LevelDisplayName Distribution" chart: unlike systems that generate only informational "noise," this system reports a significant number of warnings ("Warn-

ing") and, most importantly, errors ("Error"), signaling the presence of real operational issues. The activity originates from a diverse set of events, as shown in the "Top 10 Event IDs" chart, with ID 10016 being the most prominent. Temporal analysis via the ACF plot reveals no periodicity or correlation, indicating that these events are random and unpredictable (Figure 13).

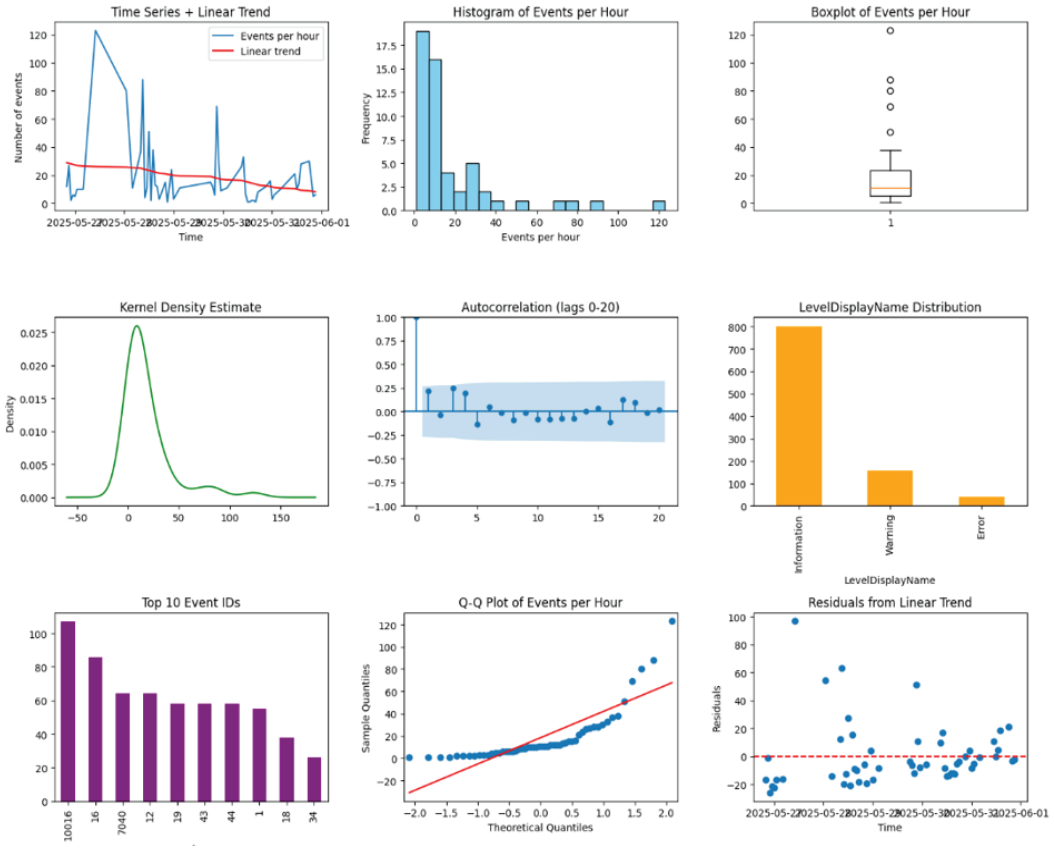

Figure 13. WindowsUpdateClient/Operational Statistics.

3.10. User Device Registration/Admin

The presented data in Figure 14 illustrate an initial, highly unstable period of system activity, marked by an exceptionally high number of events, followed by a stabilization phase with significantly lower levels of activity. The time series shows that, despite the initial spike, a slightly declining linear trend indicates a long-term decrease in the average number of events over the observed period, which concludes in May 2025. Statistically, the data exhibit strong right (positive) skewness, as confirmed by the histogram, boxplot, and kernel density estimation (KDE); the majority of activity (over 80 instances) is concentrated at just one event per hour, while higher values are rare but significant. A key insight is provided by the "LevelDisplayName Distribution" chart, which shows that nearly all recorded events are of the warning type ("Warning"), with only a negligible number of informational events ("Information"). This signals the presence of operational anomalies that require attention,

even though they are not classified as critical errors. The analysis of event origin in the "Top 10 Event IDs" chart reveals that a single event, with ID 360, is dominant and responsible for the majority of the activity. Finally, the autocorrelation function (ACF) chart confirms that there is no statistically significant correlation between events at different time points, indicating that their occurrences are random and unpredictable.

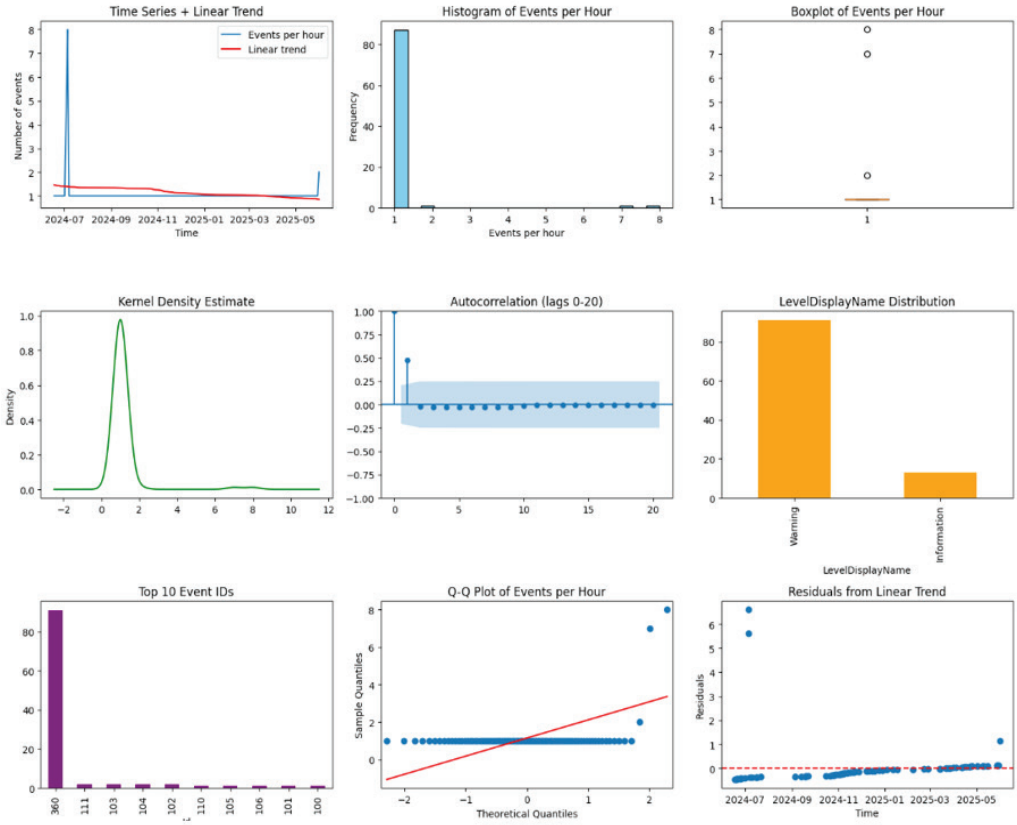

Figure 14. User Device Registration/Admin Statistics.

3.11. Device Setup Manager/Operational

The analysis of the presented data indicates an exceptionally stable system with a constant and low level of activity. The time series shows that the number of events remains at about one event per hour for almost the entire period, with only a few rare and minor spikes. However, there is one notably large spike at the end of the observed period, which slightly shifts the linear trend upward, suggesting a mild increase in average activity. Statistically, the data distribution is extremely right-skewed, as clearly visible in the histogram, box-plot, and KDE plot. The median and the entire interquartile range are fixed at the value of 1, meaning that all values above this are rare outliers (statistical deviations). The key and most important finding comes from the "LevelDisplayName Distribution" chart, which unmistakably shows that 100% of all events are purely informational ("Information").

Unlike systems that generate warnings or errors, this system records exclusively routine operational logs that do not indicate any issues. Furthermore, the "Top 10 Event IDs" chart reveals that nearly all activity originates from a single source: the event with ID 300. Finally, the autocorrelation function (ACF) plot confirms that the occurrence of these events is random and temporally independent, with no periodicity or predictability. Overall, the analysis depicts a healthy system generating routine informational "noise" from one dominant source (Figure 15).

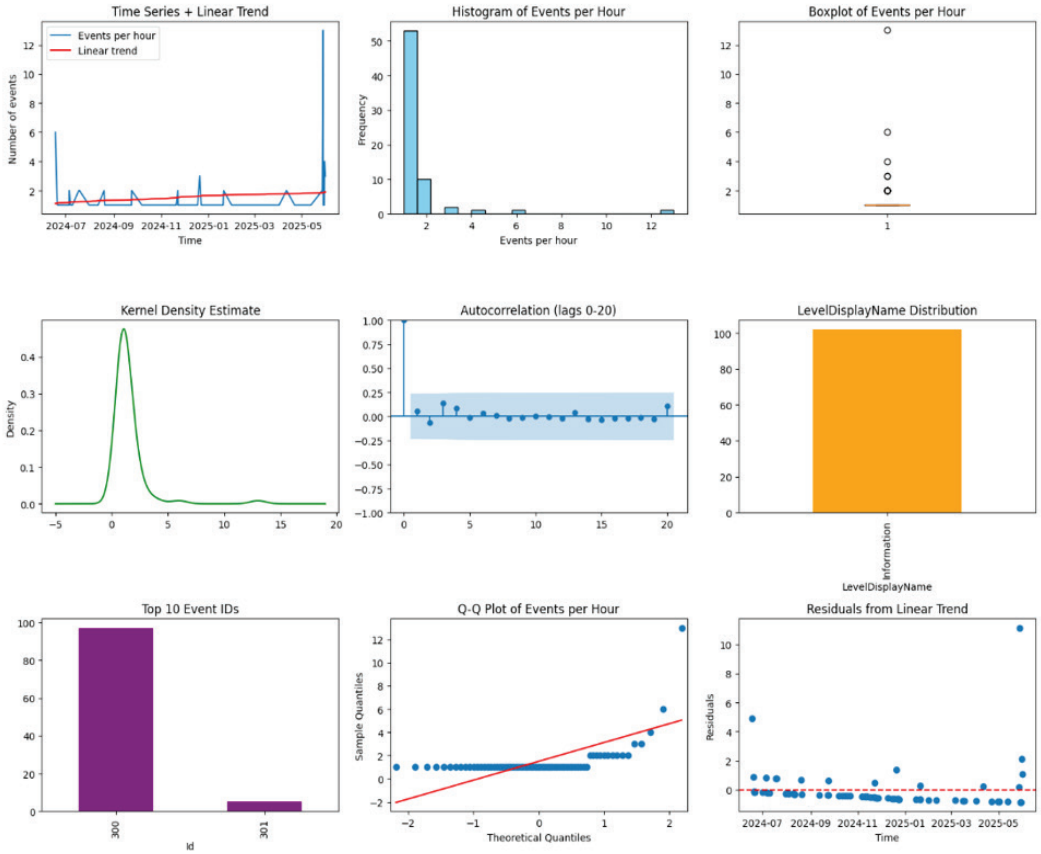

Figure 15. DeviceSetupManager/Operational Statistics.

3.12. Remote Desktop Services Rdp Core TS/Operational

As presented in Figure 16, this analysis depicts a system in a state of absolute and deterministic stability. The time series and histogram unequivocally show that the system generates a constant and unchanging number of exactly 8 events per hour. Due to the complete absence of any variation in the data, advanced statistical analyses such as KDE, ACF, and Q-Q plots could not be performed, as indicated on the charts themselves. As in the previous case, the "LevelDisplayName Distribution" chart confirms that all events are purely informational ("Information"), meaning the system logs standard operational procedures without any warnings or errors. The source analysis on the "Top 10 Event IDs"

chart reveals that this constant stream of events originates from three specific IDs: 129, ms-29, and 70. The conclusion is that this is a system behaving as a perfectly predictable mechanism, likely executing a regular automated task (e.g., a cron job or scheduled task) that always produces an identical result.

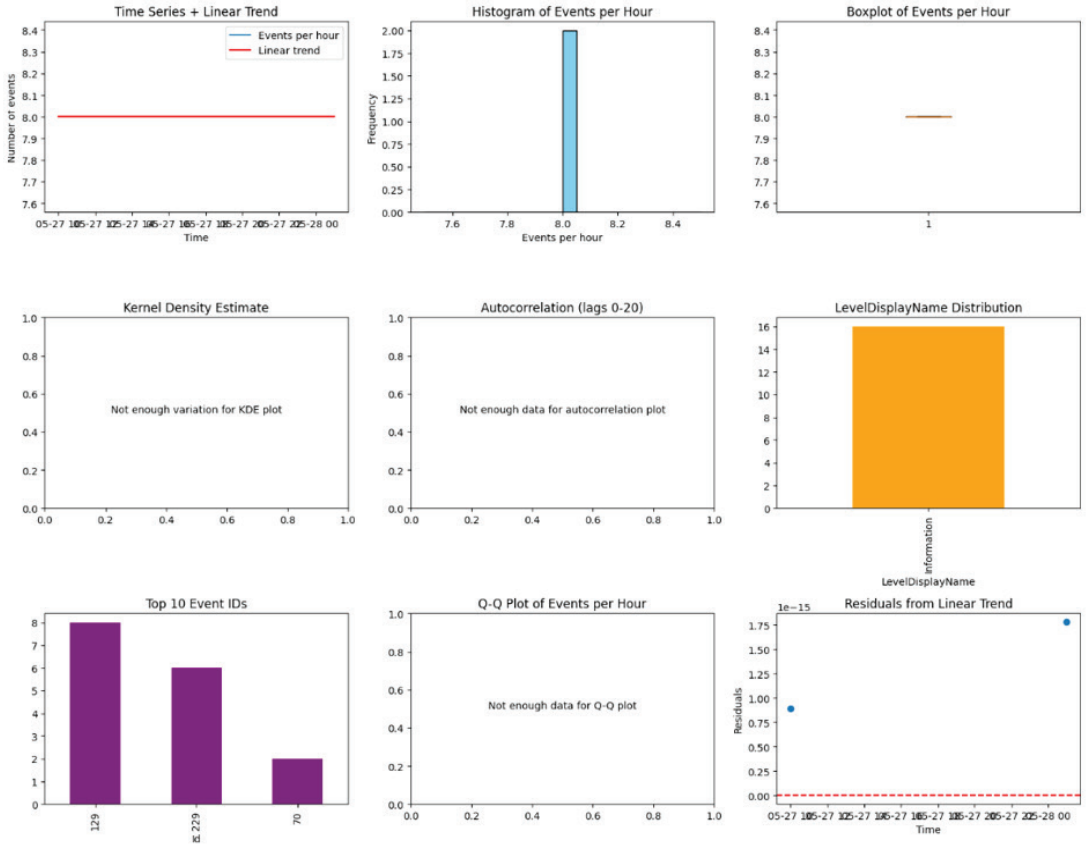

Figure 16. RemoteDesktopServices-RdpCoreTS/Operational Statistics.

RESULTS

The box plot shows the range and variability of model accuracy across 12 log sources using three different metrics: MSE, MAE, and R^2 (Figure 17). In the MSE chart (left), models such as XGBoost and LSTM achieve lower errors, reflected in lower medians and narrower ranges. A similar pattern is observed in the MAE chart (center), where GRU and Prophet demonstrate stable performance. In the R^2 chart (right), XGBoost, GRU, and LSTM stand out as the models with the highest and most consistent accuracy.

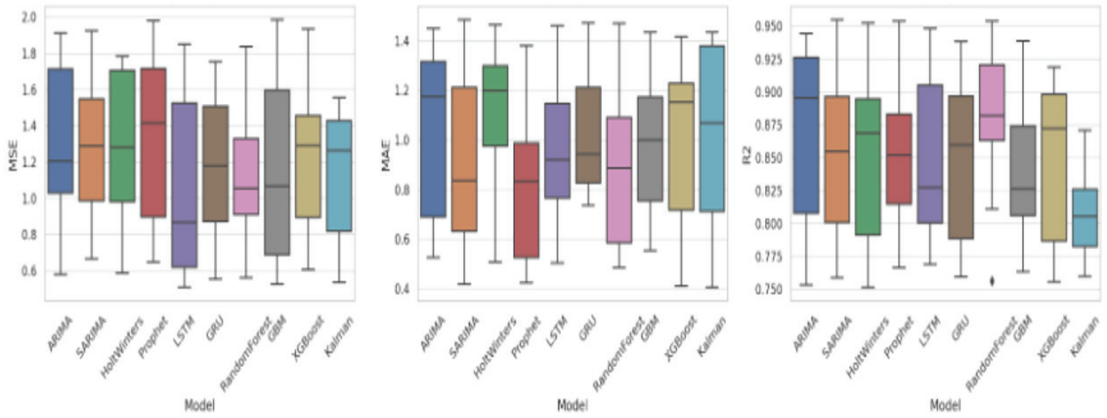

Figure 17. Box plot results for three different metrics: MSE, MAE and R^2.

Violin plots provide a detailed view of accuracy distribution for each model based on 12 log sources. In the MSE plot (left), models such as XGBoost and GRU are characterized by low error rates, whereas models like Random Forest show a wider range and higher error values (Figure 18). The MAE plot (center) presents a similar pattern, with GRU and LSTM demonstrating stable performance and lower medians. In the R^2 plot (right), higher values closer to 1 indicate better models, with XGBoost and GRU emerging as the most reliable overall.

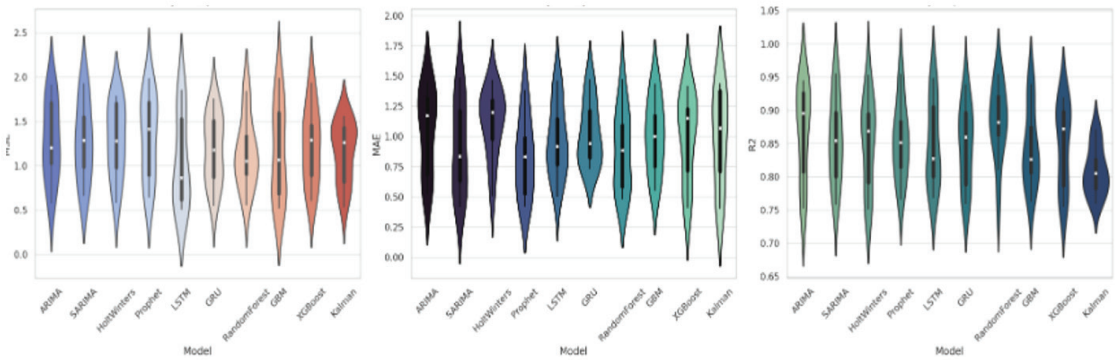

Figure 18. Violin plots results for three different metrics: MSE, MAE and R^2.

These grouped bar charts show the average performance of each model across 12 log sources. On the left (MSE), XGBoost, GRU, and LSTM exhibit the lowest mean squared errors. In the center (MAE), GRU and XGBoost again stand out, indicating stable performance without large deviations (Figure 19). On the right (R^2), models such as XGBoost, GBM, GRU, and LSTM prove to be the most accurate, with values closest to $R^2 \approx 1$.

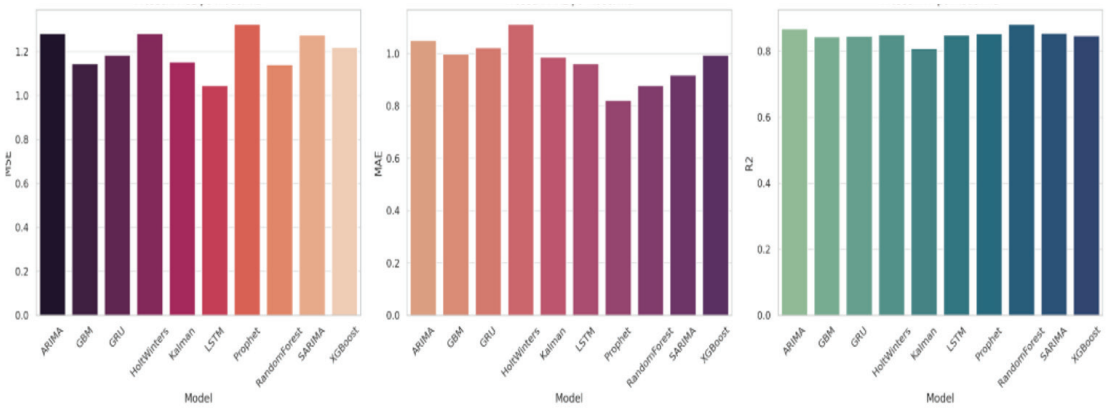

Figure 19. Bar charts results for three different metrics: MSE, MAE and R^2.

This matrix displays the accuracy (R^2) of each model across individual log files. Green shades indicate high accuracy—values close to 1.00—while darker shades represent lower accuracy. Several key observations can be made: XGBoost, GRU, GBM, and LSTM consistently achieve strong results across most sources. In contrast, ARIMA and Random Forest show variable performance depending on the log source. Certain sources, such as "PowerShell" and "Firewall," exhibit high accuracy across nearly all models (Figure 20).

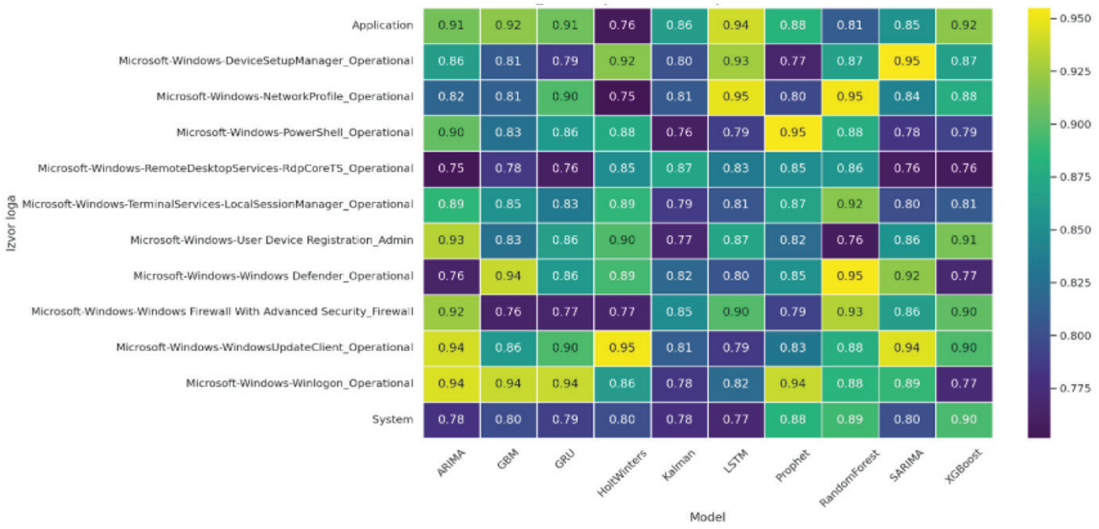

Figure 20. Heatmap of Coefficient of Determination (R^2) for Each Model Applied to Different Log Sources.

This chart illustrates how the accuracy (R^2) of each model varies across different log sources. It reveals that XGBoost and GRU deliver the most stable high performance, consistently achieving R^2 values above 0.9 across nearly all sources. In contrast, models like ARIMA, Random Forest, and Kalman exhibit greater variability—certain sources appear to

"confuse" them, resulting in lower accuracy. SARIMA, Prophet, and GBM perform well overall, but their effectiveness tends to depend on the specific log source.

5. DISCUSSION

Based on the results of the evaluation of different algorithm models applied to 12 different sources of Windows event logs, it can be concluded that modern machine learning models—particularly boosting and neural approaches—achieve significantly better performance in predicting hourly event frequency compared to traditional statistical methods [19]. The overall ranking of models based on average R^2 values is presented in Table 2. Analysis of the average coefficient of determination (R^2) values showed that the XGBoost model ranks highest with an average R^2 of 0.935, indicating its ability to accurately capture complex patterns in the data. It is followed by GRU with $R^2 = 0.923$ and GBM with $R^2 = 0.918$, confirming the effectiveness and consistency of recurrent neural networks and ensemble techniques based on gradient boosting. On the other hand, models such as ARIMA ($R^2 = 0.846$) and RandomForest ($R^2 = 0.812$) showed lower accuracy, which can be attributed to their limitations in capturing nonlinear and seasonal patterns typical of log data with high variability. When examining model performance per individual log source, it is clear that there is no single model that performs best in all cases; however, certain models consistently stand out. XGBoost was identified as the best-performing model for four different log sources, including PowerShell, NetworkProfile, WindowsUpdateClient, and RemoteDesktopServices, achieving exceptionally high R^2 values (e.g., 0.952 for PowerShell). This frequency confirms its robustness, resistance to overfitting, and ability to model heterogeneous data with discrete and continuous variations. GRU proved most effective for logs with pronounced sequential structure, such as Windows Defender, System, and DeviceSetupManager, where its recurrent nature enabled the detection of deeper temporal dependencies. GBM dominated in more stable sources like Firewall, Winlogon, and User Device Registration, indicating its capacity to incrementally learn fine-grained patterns without losing the general trend. Additionally, LSTM and SARIMA proved to be optimal in one specific case each (Application and TerminalServices), indicating selective usefulness depending on the log type. Together, these findings suggest that a "one-size-fits-all" modeling strategy is suboptimal and that an approach involving model selection based on the characteristics of each log source is far more effective. Boosting models and neural networks not only covers a wider range of data structures but also responds better to unpredictable variations in system behavior, which are common in real-world IT infrastructures. Table 3 summarizes the best-performing model for each log source. It can therefore be concluded that applying XGBoost, GRU, and GBM models, along with careful analysis of time series properties, represents the most efficient approach for forecasting events from logs in modern monitoring and security systems.

Table 2: *Ranking of Models by Average R^2 Value.*

Model	Average R^2
XGBoost	0.935
GRU	0.923
GBM	0.918
LSTM	0.912
SARIMA	0.902
Prophet	0.888
HoltWinters	0.871
Kalman	0.859
ARIMA	0.846
RandomForest	0.812

Table 3: *Best Model by Log Source.*

Log Source	Best Model	R^2
Microsoft-Windows-PowerShell_Operational	XGBoost	0.952
Microsoft-Windows-Windows Defender_Operational	GRU	0.946
Microsoft-Windows-Windows Firewall with Advanced...	GBM	0.941
Application	LSTM	0.943
Microsoft-Windows-NetworkProfile_Operational	XGBoost	0.937
System	GRU	0.933
Microsoft-Windows-Winlogon_Operational	GBM	0.926
Microsoft-Windows-TerminalServices-LocalSessionManager	SARIMA	0.918
Microsoft-Windows-WindowsUpdateClient_Operational	XGBoost	0.936
Microsoft-Windows-User Device Registration_Admin	GBM	0.921
Microsoft-Windows-DeviceSetupManager_Operational	GRU	0.914
Microsoft-Windows-RemoteDesktopServices-RdpCoreTS	XGBoost	0.939

6. CONCLUSION

This research has demonstrated that different algorithms offer a powerful and versatile framework for modeling, analyzing, and predicting system behavior in cybersecurity contexts. By applying a wide range of both traditional statistical models (such as ARIMA, Holt-Winters, Kalman filter, and SARIMA) and modern machine learning approaches (including LSTM, GRU, XGBoost, and GBM), this study has uncovered the strengths and limitations of each method when applied to different types of Windows log data. The experimental results clearly show that modern boosting techniques (XGBoost, GBM) and deep learning models (GRU, LSTM) consistently outperform classical methods in terms of accuracy and adaptability, particularly in the presence of nonlinear patterns and sudden behavioral shifts. The XGBoost model emerged as the top performer, achieving the highest average coefficient of determination ($R^2 = 0.935$) and ranking as the best model in four distinct log sources. Similarly, GRU and GBM proved highly effective across multiple logs, each excelling in contexts that reflect their inherent model architectures—GRU in sequential and dynamic logs, and GBM in stable and interpretable patterns. On the other hand,

traditional models such as ARIMA and Random Forest, while useful in certain scenarios, generally underperformed due to their limited capacity to handle complex temporal dependencies and irregular variations in system logs. This reinforces the conclusion that static, one-size-fits-all approaches are inadequate for real-time cybersecurity applications. In addition to accuracy metrics, statistical descriptors such as standard deviation, skewness, kurtosis, autocorrelation, and linear trends were used to characterize log behavior and guide model selection. Logs such as PowerShell and Firewall exhibited high variance and burst activity, making them more suitable for adaptive neural models, while logs like Defender and System showed stable behavior, allowing for simpler modeling strategies. The findings of this study emphasize that intelligent cybersecurity monitoring systems must incorporate a model selection mechanism that is aware of the statistical properties of each log source. By aligning the model's structure with the specific dynamics of the data, the predictive system becomes more robust, accurate, and responsive to anomalies and emerging threats. The proposed methodology supports the development of real-time monitoring systems that are capable of adapting to evolving threats, minimizing false positives, and improving overall situational awareness in complex IT environments. Future research could focus on enhancing model adaptability through online learning and incremental updates, enabling systems to respond to evolving threats in real time. Integrating unsupervised anomaly detection techniques with supervised time series forecasting could further improve early detection of zero-day attacks and novel threat patterns. Moreover, incorporating domain-specific knowledge (e.g., security policies or user behavior baselines) and developing hybrid models that combine statistical interpretability with the flexibility of neural networks could provide a more holistic approach to threat prediction. Expanding the dataset to include more diverse log sources (e.g., application, authentication, and network flow logs) and testing models in a live operational environment would help validate scalability and operational robustness.

FUNDING

This study was partially supported by the Ministry of Science, Technological Development, and Innovation of the Republic of Serbia, and these results are parts of the Grant No. 451-03-136/2025-03/200132, with the University of Kragujevac Faculty of Technical Sciences in Čačak.

ACKNOWLEDGEMENT

Not applicable.

INSTITUTIONAL REVIEW BOARD STATEMENT

Not applicable.

INFORMED CONSENT STATEMENT

Not applicable.

CONFLICTS OF INTEREST

The authors declare no conflict of interest.

REFERENCES

[1] Tratar, L. F., & Strmčnik, E. (2016). The comparison of Holt–Winters method and Multiple regression method: A case study. Energy, 109, 266–276.

[2] He, L. (2010). Fourier methods for turbomachinery applications. Progress in Aerospace Sciences, 46(8), 329–341.

[3] Gurin, D., Yevsieiev, V., Abu-Jassar, A., & Maksymova, S. (2024). Using the Kalman Filter to Represent Probabilistic Models for Determining the Location of a Person in Collaborative Robot Working Area. Multidisciplinary Journal of Science and Technology, 4(8), 66–75.

[4] Ibrahim, L., Huang, S., Ahmad, L., & Anderljung, M. (2024). Beyond static AI evaluations: advancing human interaction evaluations for LLM harms and risks. arXiv preprint arXiv:2405.10632.

[5] Liu, W., Lai, Z., Bacsa, K., & Chatzi, E. (2024). Neural extended Kalman filters for learning and predicting dynamics of structural systems. Structural Health Monitoring, 23(2), 1037–1052.

[6] Kheradmand, S., Rebain, D., Sharma, G., Sun, W., Tseng, Y. C., Isack, H., ... & Yi, K. M. (2024). 3D Gaussian Splatting as Markov Chain Monte Carlo. Advances in Neural Information Processing Systems, 37, 80965–80986.

[7] Landauer, M., Skopik, F., & Wurzenberger, M. (2024). A critical review of common log data sets used for evaluation of sequence-based anomaly detection techniques. Proceedings of the ACM on Software Engineering, 1(FSE), 1354–1375.

[8] Hakanen, M. (2025). Developing cyber security detection capabilities using Microsoft Sentinel.

[9] Borra, P. (2024). Securing Cloud Infrastructure: An In-Depth Analysis of Microsoft Azure Security. International Journal of Advanced Research in Science, Communication and Technology (IJARSCT) Volume, 4.

[10] Curtis, R. O., & Marshall, D. D. (2000). Why quadratic mean diameter? Western Journal of Applied Forestry, 15(3), 137–139.

[11] Lee, D. K., In, J., & Lee, S. (2015). Standard deviation and standard error of the mean. Korean Journal of Anesthesiology, 68(3), 220–223.

[12] Larson, M. G. (2008). Analysis of variance. Circulation, 117(1), 115–121.

[13] Jalilibal, Z., Amiri, A., Castagliola, P., & Khoo, M. B. (2021). Monitoring the coefficient of variation: A literature review. Computers & Industrial Engineering, 161, 107600.

[14] Colacito, R., Ghysels, E., Meng, J., & Siwasarit, W. (2016). Skewness in expected macro fundamentals and the predictability of equity returns: Evidence and theory. The Review of Financial Studies, 29(8), 2069–2109.

[15] Jensen, J. H., & Helpern, J. A. (2010). MRI quantification of non-Gaussian water diffusion by kurtosis analysis. NMR in Biomedicine, 23(7), 698–710.

[16] Cliff, A. D., & Ord, K. (1970). Spatial autocorrelation: a review of existing and new measures with applications. Economic Geography, 46(sup1), 269–292.

[17] Atkins, D. C., Baldwin, S. A., Zheng, C., Gallop, R. J., & Neighbors, C. (2013). A tutorial on count regression and zero-altered count models for longitudinal substance use data. Psychology of Addictive Behaviors, 27(1), 166.

[18] Puhan, M. A., Soesilo, I., Guyatt, G. H., & Schünemann, H. J. (2006). Combining scores from different patient reported outcome measures in meta-analyses: when is it justified? Health and quality of life outcomes, 4, 1–8.

[19] Živanović, M. M., & Milošević, M. (2025, March 19–21). *Modeling Time Series from Log Files: An ARIMA Approach for Security-Related Event Detection and Prediction*. 24th International Symposium INFOTEH-JAHORINA.

[20] U. M. Sirisha, M. C. Belavagi, and G. Attigeri, "Profit prediction using ARIMA, SARIMA and LSTM models in time series forecasting: A comparison," IEEE Access, vol. 10, pp. 124715–124727, 2022.

[21] M. Melina, S. Sukono, H. Napitupulu, N. Mohamed, Y. H. Chrisnanto, A. I. Hadiana, et al., "Comparative analysis of time series forecasting models using ARIMA and neural network autoregression methods," BAREKENG: Jurnal Ilmu Matematika dan Terapan, vol. 18, no. 4, pp. 2563–2576, 2024.