IJECBE (2026), **4**, **1**, 165–201
Received (11 May 2026) / Revised (22 June 2026)
Accepted (23 June 2026) / Published (30 March 2026)
https://doi.org/10.62146/ijecbe.v4i1.243
https://ijecbe.ui.ac.id
ISSN 3026-5258

International Journal of Electrical, Computer and Biomedical Engineering

RESEARCH ARTICLE

A Hybrid Endpoint-Network Correlation Framework for Incident-Oriented Analysis

Clavincy Francis Yohanes Ngantung and I Gde Dharma Nugraha*

Department of Electrical Engineering, Faculty of Engineering, Universitas Indonesia, Depok, Indonesia
*Corresponding author. Email: i.gde@ui.ac.id

Abstract

Security incidents often leave related traces across endpoint and network sources, but these traces may remain fragmented when each source is analyzed separately. This paper presents a hybrid endpoint-network correlation framework for incident-oriented analysis. The framework combines rule-based evidence and machine-learning-supported suspiciousness to retain candidate endpoint events and network flows, then correlates them using temporal proximity, entity consistency, and behavioral relevance. The evaluation uses three public SAPPAN host-network scenarios: Drupal vulnerability scenario-successful, Samba file-sharing known-creds, and Samba file-sharing reconnaissance. In the Drupal scenario, the framework retained 14 endpoint candidates and 3 network candidates, producing 8 hard links and 32 soft links. In the Samba known-creds scenario, it retained 4 endpoint candidates and 3 network candidates, producing 8 hard links and 4 soft links. In the reconnaissance scenario, no endpoint candidates were retained, while 2011 network flows were preserved as network-dominant evidence, resulting in a network-only context. These results show that the framework can organize retained evidence into linked or one-sided incident context without forcing unsupported endpoint-network relations. The study remains limited to scenario-level validation and does not claim event-level detection performance.

Keywords: endpoint-network correlation, incident-oriented analysis, security event correlation, endpoint-evidence, network flow analysis, hybrid security analysis, machine-learning-supported suspiciousness, scenario-level evaluation

1. Introduction

Security incidents are often difficult to interpret when endpoint and network evidence are analyzed separately. Endpoint logs may show process execution, command-line activity, user context, or host-level behavior, while network flows may show service

access, communication direction, protocol use, and scanning patterns. Each source provides useful evidence, but each source also gives only a partial view of the incident. A suspicious command may need network context to explain its role, while suspicious traffic may remain unclear without host-side activity [1], [2], [3], [4].

This problem is relevant beyond a single attack type. Previous ransomware studies show that runtime behavior and host-side traces can reveal suspicious activity beyond static signatures [5], [6], [7]. Broader security studies also show that system logs and flow-level traffic can support anomaly detection and incident analysis from different perspectives [1], [2], [3], [4]. However, when these sources are handled independently, the result is often a set of isolated suspicious records rather than a connected incident context.

Correlation is therefore important for incident-oriented analysis. In network forensics and security monitoring, correlation has been used to connect heterogeneous observations and support incident understanding [8], [9]. Context-oriented approaches such as DEEPCASE [10] and HOLMES [11] also show that relationships between events can be as important as individual event scores. Recent work on security event correlation further highlights that linking evidence across different sources, timestamps, and formats remains a practical challenge [12].

This paper proposes a hybrid endpoint-network correlation framework for incident-oriented security analysis. The term hybrid refers to the use of both rule-based indicators and machine-learning-supported suspiciousness in the endpoint-side and network-side analysis layers. Rule-based analysis is used to capture explicit and explainable behavior patterns, while machine learning is used only as a supporting layer to preserve additional suspicious candidates. The ML output is not treated as a final attack classification result.

The framework works in three analytical stages. First, endpoint-side analysis retains candidate host events based on rule evidence or ML-supported suspiciousness. Second, network-side analysis retains candidate flows that show relevant service, communication, or scan-like behavior. Third, the retained endpoint and network candidates are correlated through temporal proximity, entity consistency, and behavioral relevance. This design avoids correlating all raw records directly and instead focuses the correlation process on evidence that already carries analytical value.

The evaluation uses three public host-network scenarios from the SAPPAN dataset: Drupal vulnerability scenario-successful, Samba file-sharing known-creds, and Samba file-sharing reconnaissance [13], [14]. These scenarios provide different evidence distributions. Drupal and Samba known-creds contain both endpoint and network evidence, while Samba reconnaissance is dominated by network-side scanning behavior. Since these scenarios do not represent complete ransomware execution, this paper is framed as incident-oriented endpoint-network correlation rather than ransomware-specific detection.

The ground truth is also used with caution. SAPPAN provides scenario-level descriptions and expected evidence, not complete event-level labels for every endpoint record, network flow, or endpoint-network pair [13], [14]. For this reason, the framework is not evaluated as a binary classifier using accuracy, precision, recall, F1-score, or ROC-AUC. Instead, the evaluation focuses on candidate preservation, rule-

only/ML-only/hybrid contribution, endpoint-network correlation, and alignment with scenario-level evidence.

The scope of this paper is intentionally limited. The framework is interpreted as an incident-oriented correlation approach for organizing retained endpoint and network evidence into linked or one-sided context. It does not claim universal detection coverage or universally valid thresholds for all operational environments. The main purpose is to examine how rule-based evidence, ML-supported suspiciousness, and endpoint-network correlation can work together under controlled public scenarios.

The contributions of this paper are summarized as follows. First, it presents a staged hybrid framework in which endpoint and network layers produce candidate evidence rather than final alert verdicts. Second, it clarifies the role of machine learning as a suspiciousness-support layer, not as the main decision engine. Third, it defines a correlation mechanism based on temporal proximity, entity consistency, and behavioral relevance. Fourth, it separates hard links, soft links, unsupported relations, and network-only context so that cross-source relations are not forced when the available evidence is asymmetrical.

2. Related Work

2.1 Behavioral and Endpoint-Side Analysis

Behavioral analysis has been widely used in security research because many attacks leave traces through runtime activity, even when static indicators are incomplete. In ransomware studies, Kharraz et al. showed that malicious behavior can be observed from interactions with user data and desktop artifacts [5]. Sgandurra et al. also demonstrated that dynamic analysis combined with machine learning can capture behavioral patterns across ransomware families

[6]. These studies show that host-side behavior is useful as evidence, especially when the analysis needs to look beyond signatures.

Endpoint-side analysis extends this idea by using logs, process activity, command execution, and user context to understand what happens on a host. DeepLog, for example, shows that system logs can be modeled to detect abnormal sequences [2], while host-behavior studies indicate that endpoint records can reveal activity patterns relevant to security analysis [3]. However, endpoint evidence alone does not always explain the full incident. A suspicious command or process may still need network context to show whether it is connected to service access, scanning, exploitation, or follow-on communication.

2.2 Network-Side Security Analysis

Network-side analysis provides another view by focusing on communication behavior. Flow-based monitoring can represent traffic through features such as source and destination addresses, ports, protocol, duration, bytes, and packets [1]. These features are useful for observing service interaction, scanning behavior, or unusual communication patterns. Studies on network intrusion datasets also show that network traffic can support security analysis across different attack conditions [4], [14].

Even so, network evidence has its own limitation. A flow can show that communication happened, but it may not identify the host-side action behind it. This becomes

important in incident analysis because traffic patterns may have different meanings depending on the endpoint context. For example, service access over SMB may be more meaningful when it is connected with authentication or command-related traces, while scanning traffic may need to be treated as network-dominant evidence when endpoint support is weak. This is why endpoint and network analysis are better treated as complementary layers.

2.3 Hybrid Analysis and Event Correlation

Hybrid analysis is useful because rule-based and machine-learning-based methods serve different roles. Rule-based detection can capture known behavior patterns in an explainable way, while machine learning can provide suspiciousness scoring for records that may not be strongly captured by rules. In this study, machine learning is not used as the final decision engine. It only supports candidate preservation before correlation, which follows the framework design used in the thesis.

Correlation is needed because incident evidence is often fragmented. Raftopoulos et al. showed that network forensics depends on connecting heterogeneous records to support incident diagnosis [8]. Špaček and Čeleda also showed that correlating flows and logs can improve threat detection by combining different monitoring perspectives [9]. Context-oriented systems such as DEEPCASE and HOLMES further highlight that relationships among events can be important for understanding an attack sequence, not only the individual event score [10], [11]. Recent survey work also confirms that multi-source event correlation remains difficult because security data may differ in source, format, timing, and meaning [12].

For this reason, correlation should not be treated as a simple record join. A useful correlation process needs to consider whether endpoint and network observations are close in time, whether they share related entities, and whether their behaviors make sense together. These dimensions are important so that the framework can form linked context when evidence supports it, while still allowing network-only or unsupported conditions when the evidence is asymmetrical.

2.4 Research Gap

Previous studies provide strong foundations for behavioral analysis, endpoint analytics, network-flow analysis, and event correlation. However, many of them focus on one side of the evidence or one detection objective. Endpoint-focused studies may identify suspicious host behavior, but they do not always connect it with communication evidence. Network-focused studies can reveal traffic patterns, but they may not explain the host-side activity behind the traffic. Correlation-oriented studies address fragmented evidence, but they do not always start from a staged endpoint-side and network-side candidate-preservation process.

This study is positioned in that gap. It does not propose a new machine-learning algorithm or claim a universal detection model. Instead, it combines rule-based indicators and machine-learning-based suspiciousness scoring to preserve candidate evidence from endpoint and network data. These candidates are then correlated using temporal proximity, entity consistency, and behavioral relevance. The intended

contribution is a controlled incident-context construction process, not a claim of general detection superiority.

Table 1. Comparative Position of Prior Work

Study	Main Focus	Evidence Source	Evaluation / Dataset Scale	Correlation Granularity	Reported Output	Position Relative to This Study
Kharraz et al. [5]	Automated ransomware behavior analysis	Endpoint-side runtime behavior	Large-scale ransomware focused dynamic analysis	No endpoint-network correlation	Ransomware behavior detection	Provides ransomware behavior background, but does not focus on cross-source incident correlation
Sgandurra et al. [6]	Dynamic ransomware analysis using machine learning	Endpoint-side behavioral traces	Ransomwar family-oriented analysis	No endpoint-network correlation	ML-based ransomware detection	Shows the value of behavioral ML, but remains focused on ransomware detection rather than incident-context construction
Hofstede et al. [1]	Flow monitoring and network traffic analysis	Network flows	Network monitoring and flow-analysis perspective	Flow-level analysis only	Flow-based traffic visibility	Provides network-flow foundation, but does not link flows with endpoint-side activity
DeepLog [2]	Log anomaly detection using deep learning	System logs	System-log sequence modeling	Event-sequence modeling within logs	Anomaly detection and diagnosis	Supports log-based anomaly analysis, but does not address endpoint-network evidence linkage
Raftopoulos et al. [8]	Log correlation for network forensics	Heterogene forensic logs	Network-forensicsori analysis	Cross-log forensic correlation	Incident diagnosis support	Supports the need for correlation, but does not define the staged endpoint-network candidatepreservation design used in this study
Špaček and Čeleda [9]	Threat detection through flow and log correlation	Network flows and logs	Flow-log correlation experiment	Crosssource correlation	Threat detection support	Closely related in motivation, but this study further separates rule/ML candidate preservation and hard/soft relation output
DEEPCASE [10]	Contextual analysis of security events	Security event sequences	Event-context learning setting	Contextual event sequence analysis	Semisupervised event interpretation	Shows the importance of event context, but does not directly focus on paired endpoint-network correlation for the evaluated SAPPAN scenarios
HOLMES [11]	Correlation of suspicious information flows for APT detection	Host and informa-tionflow evidence	APT-oriented correlation setting	Suspicious informa-tionflow correlation	Multi-stage attack detection context	Strong correlation-oriented reference, but its objective and evidence model differ from the lightweight scenario-level endpoint-network correlation in this study

Levshun and Kotenko [12]	Survey of AI techniques for security event correlation	Multi-source security events	Literature survey	General securityevent correlation models	Taxonomy, challenges, and opportunities	Establishes that multi-source event correlation remains challenging and motivates the need for interpretable correlation design
This study	**Incidentorie end-point-network correlation**	**Endpoint events and network flows**	**Three public SAPPAN host-network scenarios**	**Pair-level end-point-network relation with hard, soft, nolink, and networkonly outputs**	**Scenariolevel incident context**	**Focuses on preserving candidate evidence and organizing it into linked or onesided incident context without claiming universal detection performance**

Prior work shows that endpoint behavior, network-flow analysis, and security-event correlation each provide useful but different views of an incident. Ransomware-oriented studies such as Kharraz et al. [5] and Sgandurra et al. [6] show the importance of behavioral evidence, while flow-monitoring work explains how communication-level traces can support network visibility [1]. Log and event-correlation studies further show that relationships between records are important for incident interpretation [8], [9], [12]. Context-oriented systems such as DEEPCASE [10] and HOLMES [11] are especially relevant because they demonstrate that security events should not be interpreted only as isolated records.

However, the objective of this study is narrower than those systems. This paper does not claim to replace existing event-correlation frameworks or provide a universal detection model. Instead, it focuses on a staged endpoint-network correlation process for the available SAPPAN scenarios. Endpoint and network records are first preserved as candidates through rule-based evidence and ML-supported suspiciousness. The retained candidates are then correlated through temporal proximity, entity consistency, and behavioral relevance. The main distinction is that the framework explicitly separates stronger links, weaker links, unsupported pairs, and network-only context. This is important because the evaluated scenarios do not always contain balanced endpoint and network evidence.

3. Proposed Method

3.1 Proposed Analytical Design

The proposed method is designed as a staged analytical framework for constructing incident context from endpoint and network evidence. The input to the analytical stage consists of endpoint events and network flows from the evaluated SAPPAN scenarios [13], [14]. Instead of treating endpoint or network evidence as sufficient on its own, the framework analyzes both sources separately and then correlates the retained candidates. This design follows the idea that incident understanding depends not only on detecting suspicious records, but also on explaining how those records relate to one another across different evidence sources [8], [9], [12].

The framework consists of three main analytical stages. The first stage performs

endpoint-side analysis by examining host-level evidence, such as command activity, process-related context, user information, and event timing. The second stage performs network-side analysis by examining flow-level behavior, including service access, communication direction, port usage, protocol, packet and byte characteristics, and scan-like patterns. In both stages, rule-based indicators and machine-learning-based suspiciousness support are used together to retain candidate evidence. Rule-based indicators provide explicit and explainable signals, while machine learning is used only to support suspiciousness scoring and candidate preservation.

The output of the endpoint and network stages is not treated as a final alert verdict. Instead, each stage produces a set of candidates: endpoint candidates and network candidates . This distinction is important because the purpose of the framework is not to classify each record as malicious or benign. The candidates are passed into the correlation stage, where their relationships are examined more carefully. In this way, rule-based and machine-learning outputs become inputs for incident-context construction, not the final conclusion of the analysis.

The correlation stage is the core of the proposed method. It evaluates possible relationships between and using temporal proximity, entity consistency, and behavioral relevance. Temporal proximity considers whether endpoint and network observations occur close enough in time to support a relation. Entity consistency checks whether both sides share or align with relevant attributes, such as host, user, IP address, service port, or victim context. Behavioral relevance examines whether the meaning of an endpoint event is consistent with the communication behavior observed in the network layer. These three dimensions allow the framework to form linked context when the evidence supports it, while avoiding forced relations when the evidence is weak or asymmetrical.

The general design of the proposed analytical framework is shown in Fig. 1. Endpoint evidence and network evidence are processed in parallel through rule-based and machine-learning-based analysis. The retained candidates are then correlated to produce linked context, network-only context, endpoint-only context, or unsupported relations. This structure keeps the analysis focused on evidence preservation and correlation, while limiting the interpretation of the results to the evaluated scenarios.

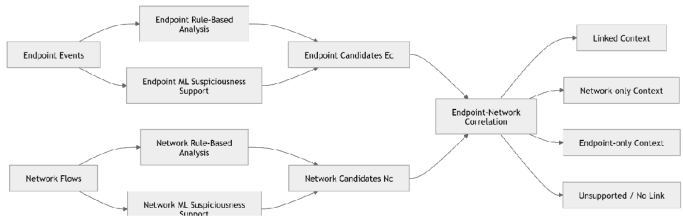

Figure 1. Proposed analytical design of the endpoint-network correlation framework

A compact representation of the analytical flow is given in Algorithm 1. The algorithm shows that candidate evidence is first preserved separately on the endpoint and network sides before correlation is performed. The correlation result is then

interpreted as context, not as a universal detection verdict.

3.2 Scenario Role and Leakage Control

The evaluation uses three public host-network scenarios from the SAPPAN dataset: Drupal vulnerability scenario-successful, Samba file-sharing known-creds, and Samba file-sharing reconnaissance [13], [14]. These scenarios are not split randomly at the row level. Instead, each scenario is assigned a functional role in the experiment. This is done because endpoint events and network flows from the same incident are related by time, entity, and activity context. If rows from the same scenario were randomly mixed across development and testing, the evaluation could unintentionally leak incident context between stages.

In this study, the Drupal vulnerability scenario-successful is used as the validation/development scenario. It is used to check the analytical flow and operational settings before the framework is evaluated on other scenarios. The Samba file-sharing known-creds and Samba file-sharing reconnaissance scenarios are used as final evaluation scenarios. These two scenarios are kept separate from development because they represent different evidence distributions: Samba known-creds contains endpoint and network evidence related to credential-based service access, while Samba reconnaissance is dominated by network-side scanning behavior.

The ground truth is used only after the framework has produced its outputs. The SAPPAN annotations provide scenario-level descriptions and expected evidence, but they do not provide a malicious or benign label for every endpoint event, network flow, or endpoint-network pair. For this reason, the ground truth is not used to tune rule decisions, train the machine-learning model, or force correlation links. It is used only to compare whether the final retained candidates and context type are consistent with the expected evidence of each scenario.

Table 2. Scenario Role and Leakage-Control Policy

Scenario	Activity Context	Evidence Orientation	Experimental Role	Leakage-Control Policy
Drupal vulnerability scenario-successful	Web-based exploitation	Endpoint + Network	Validation / development	Used to check pipeline behavior and operational settings before final evaluation
Samba file-sharing known-creds	Credential abuse / SMB-related access	Endpoint + Network	Final evaluation	Not used for training or post-result threshold adjustment
Samba file-sharing reconnaissance	Reconnaissance / port-scan behavior	Network-dominant	Final evaluation	Not used for training or post-result threshold adjustment

The role separation is also applied to the machine-learning and correlation components. Machine learning is trained as an unsupervised support layer and does not use scenario labels as targets. The correlation stage also does not read ground-truth annotations when assigning hard links, soft links, or network-only context. This keeps the analysis sequence clear: first, the framework produces candidates and correlation output; then, the output is compared with the scenario-level evidence.

Script Code 1. Scenario Role and Anti-Leakage Policy

```
DATASET_ROLES = {
    "validation_development": [
        "drupal_vulnerability_scenario-successful"
    ],
    "test_final_evaluation": [
        "samba-file-sharing-known-creds",
        "samba-file-sharing-reconnaissance"
    ]
}

ANTI_LEAKAGE_POLICY = {
    "use_ground_truth_in_rule_scoring": False,
    "use_ground_truth_in_ml_scoring": False,
    "use_ground_truth_in_correlation": False,
    "use_test_scenarios_for_training": False
}
```

This split does not remove the limitation that the evaluation is still based on three public scenarios. However, it makes the experimental boundary explicit. The Drupal scenario is not claimed as an independent test case, while the two Samba scenarios are used to observe how the framework behaves on different evidence conditions after the analytical settings have been fixed.

3.3 Rule-Based Candidate Scoring

Rule-based scoring is used as the first evidence layer in the proposed framework. Its purpose is not to decide whether an incident has occurred, but to retain endpoint events and network flows that contain explicit technical signals. This layer is kept explainable: every retained record carries a heuristic score, an activity category, and an evidence reason. This is important because the next stages do not only need suspicious records, but also need to know why those records are worth correlating.

The rule families are defined from common security analysis patterns, such as command execution, scripting activity, web execution context, service probing, SMB-related communication, and scan-like behavior. On the endpoint side, rules inspect fields such as command line, message, event original, process image, host, user, and event type. On the network side, rules inspect flow attributes such as source and destination IP address, source and destination port, protocol, bytes, packets, duration, and repeated service access. This follows the general idea that host logs and network flows provide different but complementary evidence for incident analysis [1], [8], [9].

For each rule γ in the rule set R, the matching function is defined as:

$$m_{\gamma}(x) = \begin{cases} 1, & \textit{if record } x \textit{ satisfies rule } \gamma \\ 0, & \textit{otherwise} \end{cases} \tag{1}$$

Each rule has a weight W_{γ}, which represents the strength of its heuristic evidence. The rule-based score $P_H(x)$ is taken from the strongest matching rule:

$$p_H(x) = \max_{\gamma \in R} \left(w_{\gamma} . m_{\gamma}(x)\right) \tag{2}$$

The dominant rule is then selected as:

$$\gamma^* = arg\max_{\gamma \in R} \left(w_{\gamma} . m_{\gamma}(x)\right) \tag{3}$$

The activity category and evidence reason are taken from this dominant rule:

$$category_H(x) = category(\gamma^*) \tag{4}$$

$$\text{evidence}_H(x) = \text{evidence}(r^*) \tag{5}$$

This scoring design keeps the output traceable. If an endpoint event is retained, the framework can show whether it was retained because of command execution, decoding or staging behavior, web execution context, reconnaissance trace, or another explicit signal. If a network flow is retained, the framework can also show whether the reason is SMB-related traffic, service probing, or scan-like communication.

Endpoint and network rule candidates are selected using slightly different gates. For endpoint events, a record is retained when its heuristic score reaches the endpoint threshold:

$$\text{candidate}_H^{ep}(x) = \begin{cases} 1, & p_H(x) \geq \theta_{ep}, \\ 0, & p_H(x) < \theta_{ep} \end{cases} \tag{6}$$

In the implementation, $\theta_{ep} = 0.70$. This value is not interpreted as accuracy. It is only an operational boundary indicating that the endpoint event contains sufficiently explicit evidence to be passed into the next stage.

For network flows, a record is retained when it receives any positive heuristic score:

$$\text{candidate}_H^{net}(f) = \begin{cases} 1, & p_H(f) > 0 \\ 0, & p_H(f) = 0 \end{cases} \tag{7}$$

This is used because network rules assign positive scores only when a flow matches a defined communication pattern. However, a positive score does not mean that the flow is malicious. It only means that the flow carries evidence that should be preserved for later scoring and correlation.

Table 3. Rule Families and Evidence Output

Domain	Rule Family	Example Signal	Category	Evidence Output
Endpoint	Command / script execution	PowerShell, cmd.exe, encoded command	Execution	process name, argument, command pattern
Endpoint	Decoding or staging behavior	certutil decode, base64-like artifact	Execution / Suspicious	command chain and technical argument
Endpoint	Web execution context	web interpreter, web-root path, execution argument	Exploit / Web	process, path, event text
Endpoint	Reconnaissance trace	nmap, MS17-related term, SMB 445 trace	Reconnaissance	command, target, service indicator
Network	SMB-related traffic	port 445 or 139	SMB / Suspicious	source, destination, port, protocol
Network	Service probing	port 79 or service-specific probing	Suspicious / Recon	service port and host relation
Network	Scan-like behavior	many destination ports between host pairs	Recon / Scan	number of unique ports and flow pattern

The rule layer is parameterized so that technical configurations, such as service ports,web execution patterns, and scan thresholds, can be recorded and adjusted when

telemetry changes. These parameters are not adjusted after observing final evaluation results. The main operational values are summarized later in Section 3.7 to keep this subsection focused on the scoring mechanism.

Script Code 2. Compact Rule-Based Scoring Example

```
PH_THR = 0.70
SMB_PORTS = {139, 445}
NETWORK_PORT_SWEEP_THR = 25

def endpoint_rule(text, web_context=False):
    if "encodedcommand" in text or "certutil" in text:
        return 1.00, "execution", "suspicious command usage"
    if web_context and ("cmd.exe" in text or "powershell" in text):
        return 0.70, "web_execution", "web-related command execution"
    return 0.00, "none", ""

def network_rule(dst_port, unique_dst_ports):
    if dst_port in SMB_PORTS:
        return 0.70, "service_access", "SMB-related communication"
    if unique_dst_ports >= NETWORK_PORT_SWEEP_THR:
        return 0.80, "reconnaissance", "scan-like port sweep"
    return 0.00, "none", ""
```

The output of this layer becomes one of the inputs for hybrid candidate preservation. Rule-based evidence remains traceable through P_H, category, and evidence reason, while records that are not strongly captured by rules may still be considered later by the machinelearning support layer. Ground-truth annotations are not used to generate P_H, select the dominant rule, or force candidate status. They are only used after the framework output is produced, when the result is compared with the scenario-level evidence.

3.4 Machine-Learning-Based Suspiciousness Support

The machine-learning layer is used after rule-based scoring to provide additional suspiciousness support. It is not designed to replace the rule layer and is not used as the final decision engine. Its role is narrower: to retain endpoint events and network flows that may not contain strong explicit rule evidence but still appear unusual based on their textual, technical, or flow-level characteristics.

The model used in this study is Isolation Forest. This model is selected because it supports unsupervised anomaly scoring and does not require complete event-level or flowlevel labels. This is aligned with the dataset condition, since the SAPPAN scenarios provide scenario-level evidence rather than a malicious or benign label for every endpoint event, network flow, or endpoint-network pair [13], [14]. Therefore, the ML score is not interpreted as an attack probability. It is treated as an anomaly-oriented suspiciousness score used only for candidate preservation before correlation.

Endpoint and network ML use the same model family but different feature representations. This separation is necessary because endpoint evidence and network evidence carry different meanings. Endpoint data contains command text, process context, user and host information, and event descriptions. Network data contains communication attributes such as duration, bytes, packets, ports, protocol, and traffic rate. The framework therefore builds two different ML pipelines: one for endpoint events and one for network flows.

On the endpoint side, the textual representation is built by combining available activityrelated fields, such as command line, message, event original, process image,

host, user, URL, path, and event type. This endpoint text is represented as:

$$t_i^{ep} = \text{concat}\left(\text{cmd}_i, \text{message}_i, \text{original}_i, \text{process}_i, \text{host}_i, \text{user}_i, \text{type}_i\right) \tag{8}$$

Because endpoint logs often contain variable tokens such as file paths, command arguments, IP addresses, and process parameters, the text is transformed using feature hashing. Word-level hashing captures token-level patterns, while character-level hashing captures shorter technical strings, path fragments, command variations, and obfuscated tokens. The hashed representation is then reduced using Truncated SVD so that the final text vector remains compact enough for anomaly scoring [15].

$$h_i^{\text{word}} = H_{\text{word}}\left(t_i^{ep}\right) \tag{9}$$

$$h_i^{\text{char}} = H_{\text{char}}\left(t_i^{ep}\right) \tag{10}$$

$$r_i^{ep} = \text{SVD}_k\left(h_i^{\text{word}} \oplus h_i^{\text{char}}\right) \tag{11}$$

Endpoint text features are combined with auxiliary technical features. These features are not labels. They only provide simple technical context that may help the model read endpoint behavior more clearly. The auxiliary vector is defined as:

$$\begin{aligned} a_i^{ep} &= \left[\text{len}_i, \text{tok}_i, \text{url}_i, \text{powershell}_i, \text{certutil}_i, \text{finger}_i, \text{webinterp}_i\right], \\ \text{cont.} &= \left[\text{webctx}_i, \text{nmap}_i, \text{ms17}_i, \text{port445}_i, \text{netconn}_i\right]. \end{aligned} \tag{12}$$

The final endpoint feature vector is then scaled before being passed into Isolation Forest:

$$x_i^{ep} = \text{Scale}\left(a_i^{ep} \oplus r_i^{ep}\right) \tag{13}$$

The endpoint Isolation Forest produces an anomaly score. Since the decision_function of Isolation Forest gives higher values to more normal observations, the value is inverted so that a higher score represents stronger anomaly support:

$$s_i^{ep} = -IF_{ep}.\text{decision\_function}\left(x_i^{ep}\right) \tag{14}$$

The raw anomaly score is then normalized using the score distribution learned during artifact building. In this study, the endpoint score is normalized using the median and upper quantile of the training score distribution:

$$praw_i^{ep} = \text{clip}\left(\frac{s_i^{ep} - q_{50}^{ep}}{q_{99}^{ep} - q_{50}^{ep}}, 0, 1\right) \tag{15}$$

The value $p_{raw,i}^{ep}$ is not a probability of attack. It only shows how anomalous an endpoint event is compared with the learned endpoint score distribution. To avoid exporting weak MLonly records, endpoint ML uses candidate gating. An endpoint event is considered an ML candidate only if it has not already been retained by the

rule layer, contains meaningful text, is not suppressed as noise, and is not only weak network-connection telemetry without endpoint context:

$$C_i^{ep} = 1 if pH_i^{ep} < \theta_{ep} \wedge semantic_i = 1 \wedge noise_i = 0 \wedge weaknet_i = 0 \tag{16}$$

The candidate set is then ranked using percentile rank:

$$p_{rank,i}^{ep} = \text{rank\_pct}\left(p_{raw,i}^{ep} \mid C_i^{ep} = 1\right) \tag{17}$$

An endpoint event is finally marked as ML-flagged when it satisfies the rank threshold and has either enough raw anomaly support or a strong technical indicator:

$$M_i^{ep} = 1 if C_i^{ep} = 1 \wedge p_{rank,i}^{ep} \geq \tau_{\gamma ank}^{ep} \wedge \left(p_{raw,i}^{ep} \geq \tau_{raw}^{ep} \vee I_{tech,i} = 1\right) \tag{18}$$

The active endpoint ML score is only filled when the event passes this gating process:

$$pM_i^{ep} = \begin{cases} p_{rank,i}^{ep}, & M_i^{ep} = 1 \ \wedge \ p_{H,i}^{ep} < \theta_{ep} \\ \text{NaN}, & \text{otherwise} \end{cases} \tag{19}$$

This design prevents all raw anomaly scores from being treated as active ML candidates. The audit scores are still retained as pM_raw and pM_rank, but the active score pM is assigned only to endpoint events that satisfy the gating criteria.

On the network side, the model uses flow-level numerical features. The main features include bytes, packets, duration, packets per second, bytes per second, average packet size, source port, destination port, and protocol indicators. The derived flow features are defined as:

$$pps_j = \frac{pkts_j}{\max\left(duration_j, \epsilon\right)} \tag{20}$$

$$bps_j = \frac{bytes_j}{\max\left(duration_j, \epsilon\right)} \tag{21}$$

$$avgpkt_j = \frac{bytes_j}{\max\left(pkts_j, \epsilon\right)} \tag{22}$$

The final network feature vector is represented as:

$$\begin{aligned} x_j^{net} &= \text{Scale}\left(\left[bytes_j, pkts_j, duration_j, pps_j, bps_j, avgpkt_j\right]\right) \\ \text{cont.} &= \left[srcport_j, dstport_j, protoTCP_j, protoUDP_j, protoICMP_j\right] \end{aligned} \tag{23}$$

The network Isolation Forest score is computed as:

$$s_j^{net} = -IF_{net}.\text{decision\_function}\left(x_j^{net}\right) \tag{24}$$

The score is then normalized into the range 0 to 1:

$$iso\_p_j^{net} = \frac{s_j^{net} - \min(s^{net})}{\max(s^{net}) - \min(s^{net})} \tag{25}$$

Unlike endpoint scoring, the network ML layer is only allowed to flag flows that have not already been retained by the rule layer. This keeps rule-based evidence traceable and prevents ML from overwriting explicit service or scanning evidence:

$$C_j^{net} = \begin{cases} 1, & p_{H,j}^{net} = 0 \\ 0, & p_{H,j}^{net} > 0 \end{cases} \tag{26}$$

$$M_j^{net} = \begin{cases} 1, & C_j^{net} = 1 \ \wedge \ iso\_p_j^{net} \geq \tau_{net} \\ 0, & \text{otherwise} \end{cases} \tag{27}$$

The final ML outputs from the endpoint and network layers are summarized in Table 4.

Table 4. Machine Learning Features and Outputs

Domain	Feature Group	Examples	ML Process	Output
Endpoint	Text activity	command line, message, event original, process, host, user, event type	word hashing, character hashing, Truncated SVD	text representation
Endpoint	Auxiliary features	text length, token count, URL, PowerShell, certutil, Finger, web context, nmap, MS17, port 445, network connection	scaling and feature concatenation	auxiliary vector
Endpoint	Anomaly score	combined text and auxiliary vector	Isolation Forest, quantile normalization, rank gating	pM_raw, pM_rank, pM, ml_flagged
Network	Volume and duration	bytes, packets, duration	scaling	flow vector
Network	Rate features	packets per second, bytes per second, average packet size	derived feature calculation	flow vector
Network	Port and protocol	source port, destination port, TCP, UDP, ICMP	scaling and protocol encoding	iso_p, pM, ml_flagged

Table 5. Main Parameters of the Machine-Learning Layer

Parameter	Value	Domain	Function
Word hashing dimension	2048	Endpoint	Fixed-size word-level text representation
Character hashing dimension	2048	Endpoint	Fixed-size character-level text representation
Character n-gram range	3–5	Endpoint	Captures path fragments, short tokens, and command variations
SVD components	96	Endpoint	Reduces hashed text representation
Auxiliary features	12 features	Endpoint	Adds technical context to endpoint text
Network features	11 features	Network	Represents flow-level behavior
Main ML model	Isolation Forest	Endpoint and network	Unsupervised anomaly scoring
Number of estimators	200	Endpoint and network	Isolation Forest estimator count
Contamination	auto	Endpoint and network	Outlier proportion handling
Endpoint ML rank threshold	0.95	Endpoint	Percentile-rank threshold for active ML candidate
Endpoint raw floor	0.05	Endpoint	Minimum raw anomaly support
Network ML threshold	0.60	Network	Minimum normalized network anomaly score
ML fallback	Disabled by default	Endpoint	Used only for ablation or debugging
Heuristic holdout	Disabled by default	Endpoint and network	Prevents ground-truth-assisted release during scoring

The main implementation parameters used by the ML layer are listed in Table 5. These parameters are treated as operational settings and are later considered in the sensitivity discussion.

The overall ML scoring process is summarized in Algorithm 1.

Algorithm 1. Machine-Learning-Based Suspiciousness Scoring

Input : Endpoint events and network flows after rule-based scoring
Output : pM_raw, pM_rank, iso_p, ml_candidate, ml_flagged, and to_correlate

1. For each endpoint event:
   - build endpoint text from relevant fields
   - compute auxiliary technical features
   - transform text using word-level and character-level hashing
   - reduce hashed text using Truncated SVD
   - combine SVD text features with auxiliary features
   - scale the final endpoint feature vector
   - compute Isolation Forest anomaly score
   - normalize the raw score into pM_raw
   - compute percentile rank as pM_rank
   - apply semantic, noise, weak-network, rank, and raw-score gates
   - mark the event as ml_flagged if the gates are satisfied
2. For each network flow:
   - compute flow-level features from bytes, packets, duration, ports, and protocol
   - derive pps, bps, and average packet size
   - scale the network feature vector
   - compute Isolation Forest anomaly score
   - normalize the score into iso_p
   - allow ML flagging only when the flow has no active rule evidence
   - mark the flow as ml_flagged if iso_p satisfies the network ML threshold
3. Set to_correlate = rule_active OR ml_flagged
4. 4. Pass retained candidates to the hybrid candidate-preservation stage

A compact view of the ML workflow is shown in Fig. 2

A simplified implementation view is shown in Listing 2. The listing is not intended to replace the full implementation, but to make the scoring logic transparent.

Script Code 3. Simplified ML Scoring and Gating Logic

```
# Endpoint scoring
raw_ep = -endpoint_if.decision_function(X_ep)
pM_raw = normalize_by_quantile(raw_ep, q50, q99)
pM_rank = percentile_rank(pM_raw)

ml_candidate_ep = (~rule_active_ep) & semantic_gate & (~noise_gate) & (~weak_net_gate)
ml_flagged_ep = ml_candidate_ep & (pM_rank >= 0.95) & (
    (pM_raw >= 0.05) | strong_indicator
)

# Network scoring
raw_net = -network_if.decision_function(X_net)
iso_p = minmax_normalize(raw_net)

ml_candidate_net = (pH_net == 0)
ml_flagged_net = ml_candidate_net & (iso_p >= 0.60)

to_correlate = rule_active | ml_flagged
```

With this mechanism, machine learning extends candidate coverage without replacing rule-based evidence. If a record already has explicit rule evidence, the rule category and rule reason remain traceable. If a record does not meet a rule but appears anomalous and passes the gating criteria, it can still be retained as an ML-supported candidate. The retained candidates are then passed to the hybrid candidate-preservation and correlation stages, where their relation to other evidence is examined.

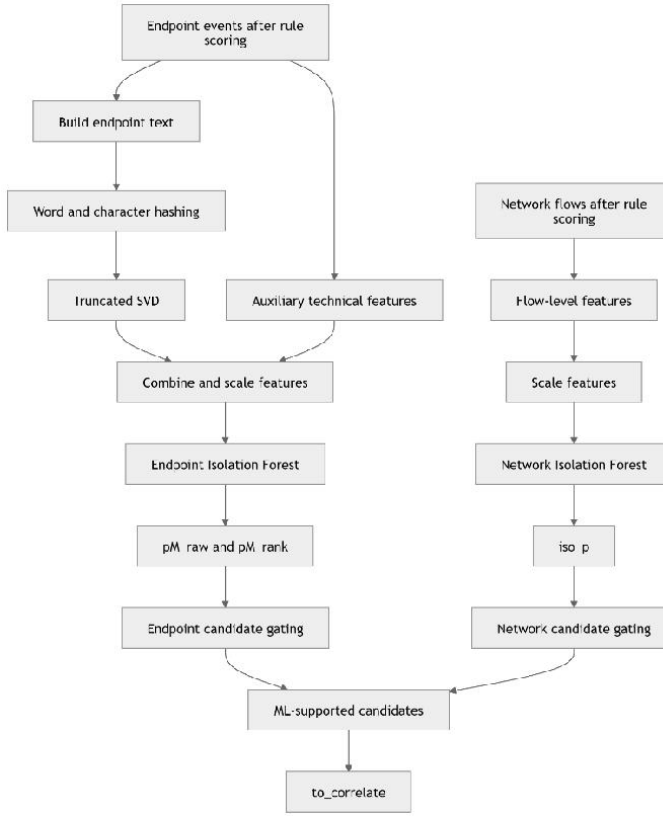

Figure 2. Machine-Learning-Based Suspiciousness Support Workflow

3.5 Hybrid Candidates Preservation

The rule-based and machine-learning layers are combined through a hybrid candidatepreservation stage. This stage is not intended to produce a final incident decision. Its role is to determine which endpoint events and network flows should be retained for endpoint-network correlation. This distinction is important because p_H and p_M come from different forms of evidence. The rule-based score p_H represents explicit heuristic evidence, while the ML score p_M represents anomaly-oriented suspiciousness support. Combining these two signals can help preserve relevant candidates, but the result still needs to be interpreted carefully because ML-based security analysis may face challenges related to noise, false positives, and generalization [15], [16].

For this reason, the framework does not combine p_H and p_M by direct summation. A summed score may suggest that both values have the same probabilistic meaning, while in practice they do not. The rule score is derived from weighted rule matches, whereas the ML score is derived from anomaly scoring and candidate gating. Therefore, this stage follows a candidate-preservation principle: a record is retained if either the rule layer or the ML layer provides active support.

The set of active scores for a record $x$is defined as:

$$P(x) = \{\, p_H(x),\, p_M(x) \mid \text{score is available, valid, and active} \,\} \tag{28}$$

The consolidated score is then written as:

$$S(x) = \max\left(P(x)\right) \tag{29}$$

More explicitly:

$$S(x) = \begin{cases} p_H(x), & p_H(x) \text{ is active } \textit{and } p_M(x) \text{ is not active} \\ p_M(x), & p_M(x) \text{ is active } \textit{and } p_H(x) \text{ is not active} \\ \max\left(p_H(x), p_M(x)\right), & p_H(x) \text{ and } p_M(x) \text{ are both active} \end{cases} \tag{30}$$

The score $S(x)$is only used as a temporary priority score for reading candidates. It is not treated as the probability that a record is malicious. If a record already has explicit rule evidence, the rule category and rule evidence are kept as the primary explanation. If a record is retained only by ML, it is marked as an ML-supported suspicious candidate. This keeps the explanation traceable and prevents ML output from overwriting explicit rule evidence, which is important for security analysis that needs interpretable evidence [17], [18].

For endpoint events, the rule-active status is defined as:

$$H_i^{ep} = \begin{cases} 1, & p_{H,i}^{ep} \geq \theta_{ep} \\ 0, & \text{otherwise} \end{cases} \tag{31}$$

The ML-active status is defined as:

$$M_i^{ep} = \begin{cases} 1, & ml\_flagged_i^{ep} = 1 \\ 0, & \text{otherwise} \end{cases} \tag{32}$$

The endpoint candidate set is then:

$$E_c = \left\{ e_i \in E \mid H_i^{ep} = 1 \ \vee \ M_i^{ep} = 1 \right\} \tag{33}$$

For network flows, the rule-active status follows the network rule output. A flow is active when it has positive heuristic evidence:

$$H_j^{net} = \begin{cases} 1, & p_{H,j}^{net} > 0 \\ 0, & \text{otherwise} \end{cases} \tag{34}$$

The network ML-active status is defined as:

$$M_j^{net} = \begin{cases} 1, & ml\_flagged_j^{net} = 1 \\ 0, & \text{otherwise} \end{cases} \tag{35}$$

The network candidate set is then:

$$N_c = \left\{ n_j \in N \mid H_j^{net} = 1 \ \vee \ M_j^{net} = 1 \right\} \tag{36}$$

The final status sent into the correlation stage is represented by to_correlate:

$$to\_correlate_i^{ep} = H_i^{ep} \ \vee \ M_i^{ep} \tag{37}$$

$$to\_correlate_j^{net} = H_j^{net} \ \vee \ M_j^{net} \tag{38}$$

This means that correlation is not applied to all raw endpoint events and network flows. It is applied only to the retained endpoint candidate set E_c and network candidate set N_c. This matters because correlation works better when the input already carries analytical relevance. In event-correlation studies, relationships between records are usually more meaningful when they are built from observations that already contain contextual, temporal, or attribute-level significance [8], [9], [12].

Table 6. Components of Hybrid Candidate Preservation

Component	Source	Function	Interpretation
p_H	Rule-based scoring	Represents explicit heuristic evidence	Not an attack probability
p_M	ML-based scoring	Represents anomaly-oriented suspiciousness support	Not a final malicious label
S_X	Active p_H and/or p_M	Keeps the strongest active support	Used for candidate priority only
ml_flagged	ML gating	Marks records that pass ML candidate criteria	Not every ML score becomes active
final_category	Rule or ML layer	Gives temporary activity category	Used for candidate reading
final_evidence	Rule or ML layer	Explains why a record is retained	Used for traceability
to_correlate	Rule-active or ML-active status	Determines whether a record is sent to correlation	Input gate for (E_c) and (N_c)

With this design, the hybrid stage acts as a controlled bridge between scoring and correlation. It does not state that a retained record is malicious. It only states that the record has enough rule-based evidence or ML-supported suspiciousness to be examined together with other retained records. The actual incident context is constructed in the next stage, where candidates are evaluated through temporal proximity, entity consistency, and behavioral relevance rather than through isolated scores alone [8], [9], [12].

3.6 Endpoint-Network Correlation

The endpoint-network correlation stage connects retained endpoint candidates and network candidates so that the analysis does not stop at two separate lists of events and flows. In incident analysis, an endpoint event may explain what happened on a host, while a network flow may show how the activity interacted with another service or host. Reading them separately can leave the incident context fragmented.

Prior work on log and security-event correlation also emphasizes that relationships between records are important for understanding incident context across heterogeneous evidence sources [8], [9], [12].

The input of this stage is not the full raw dataset. Correlation is performed only on endpoint candidates E_c and network candidates N_c that have been retained by the hybrid candidate-preservation stage:

$$E_c = \{e_1, e_2, \ldots, e_m\} \tag{39}$$

$$N_c = \{n_1, n_2, \ldots, n_k\} \tag{40}$$

This restriction is important because correlating all raw endpoint events with all raw network flows may produce many weak or incidental pairs. By using only candidates with to_correlate = True, the correlation stage starts from records that already have rule-based evidence, ML-supported suspiciousness, or both.

The correlation score is built from three main dimensions: temporal proximity, entity consistency, and behavioral relevance. Temporal proximity measures how close an endpoint event and a network flow occur in time. Entity consistency checks whether both records share or refer to related entities, such as host, IP address, victim identifier, service, or port. Behavioral relevance checks whether the endpoint-side activity and network-side communication describe compatible behavior. This follows the same general motivation used in contextual securityevent analysis, where isolated records become more useful when their relation to other evidence is considered [10], [11].

The temporal distance between endpoint event e_i and network flow n_j is defined as:

$$\Delta t(e_i, n_j) = \left| t(e_i) - t(n_j) \right| \tag{41}$$

The temporal proximity score is then computed using a decay function:

$$T(e_i, n_j) = \exp\left(decay - \frac{\Delta t(e_i, n_j)}{W} \right) \tag{42}$$

where W is the correlation time window. A smaller time difference gives a stronger temporal contribution. However, time is not used as the only basis for correlation because many unrelated activities may occur close to each other in an active system.

Entity consistency is represented as:

$$A(e_i, n_j) = \begin{cases} 1, & \text{if consistent entity evidence exists between } e_i \text{ and } n_j \\ 0, & \text{otherwise} \end{cases} \tag{43}$$

The entity evidence may come from shared or related IP addresses, host identifiers, service ports, victim-side attributes, or other fields that can reasonably connect endpoint and network evidence. This dimension prevents correlation from relying only on timestamp proximity.

Behavioral relevance is calculated from compact text representations of endpoint and network candidates. The endpoint text summarizes host-side activity, while the network text summarizes flow-side activity, such as source, destination, port, protocol, volume, and duration. Both are transformed using the linker artifact, then compared using cosine similarity:

$$z_i^{ep} = \text{normalize}\left(\text{SVD}\left(H\left(text_i^{ep}\right)\right)\right) \tag{44}$$

$$z_j^{net} = \text{normalize}\left(\text{SVD}\left(H\left(text_j^{net}\right)\right)\right) \tag{45}$$

$$R(e_i, n_j) = \text{cosine}\left(z_i^{ep}, z_j^{net}\right) \tag{46}$$

The value $R(e_i, n_j)$ is treated only as behavioral similarity. It does not prove that two records are related. For that reason, it is combined with temporal and entity evidence before a link is formed.

The final correlation score is defined as:

$$C(e_i, n_j) = w_t\, T(e_i, n_j) + w_a\, A(e_i, n_j) + w_r\, R(e_i, n_j) \tag{47}$$

where W_t, W_a, and W_γ control the contribution of temporal proximity, entity consistency, and behavioral relevance. These weights are operational design parameters, not universal constants. Their role is to combine several correlation signals into a structured score for the evaluated scenarios.

The framework also generates a relationship reason, or why, for each retained pair. This reason may come from close timestamp distance, shared or related entities, matching service context, consistent behavioral pattern, or technical evidence from the candidate records. This is important because a correlation result should not only produce a score, but also explain why the pair is kept.

The relationship between an endpoint candidate and a network candidate is divided into hard link, soft link, or no link:

$$Hard(e_i, n_j) = \begin{cases} 1, & C(e_i, n_j) \geq \tau_h \ \wedge \ Reason(e_i, n_j) = 1 \\ 0, & \text{otherwise} \end{cases} \tag{48}$$

$$Soft(e_i, n_j) = \begin{cases} 1, & \tau_s \leq C(e_i, n_j) < \tau_h \\ 0, & \text{otherwise} \end{cases} \tag{49}$$

$$NoLink(e_i, n_j) = \begin{cases} 1, & C(e_i, n_j) < \tau_s \\ 0, & \text{otherwise} \end{cases} \tag{50}$$

A hard link represents a stronger relationship because it requires both a sufficient correlation score and an explicit supporting reason. It is not formed only because two records are close in time. A soft link represents a weaker relationship that may still be useful for investigation, while no link means that the pair does not have enough support to be retained.

Hard links and soft links are made mutually exclusive for the same endpoint-flow pair. If a pair already satisfies the hard-link condition, it is not counted again as a soft link:

$$Pass(e_i, n_j) = \begin{cases} \text{hard}, & Hard(e_i, n_j) = 1 \\ \text{soft}, & Hard(e_i, n_j) = 0 \ \wedge \ Soft(e_i, n_j) = 1 \\ \text{none}, & \text{otherwise} \end{cases} \tag{51}$$

This exclusivity prevents duplicate counting and keeps the correlation output easier to inspect.

The correlation stage also avoids forcing a link when one side does not provide enough evidence. If endpoint candidates are missing or too weak, but network evidence remains meaningful, the framework can produce a network-only context. This is especially relevant for reconnaissance-oriented activity, where the strongest trace may appear in network flows rather than endpoint logs. In this case, the framework keeps the network evidence without claiming an endpoint-network relationship that is not supported by the available data.

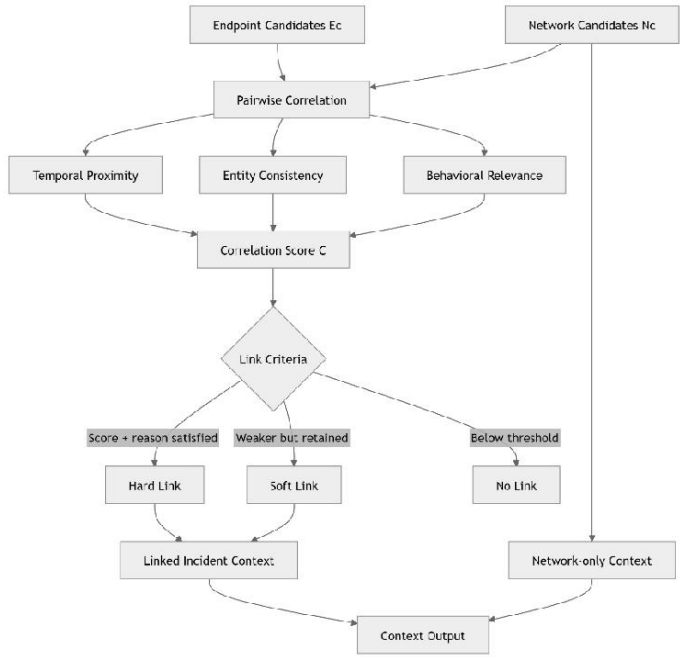

Figure 3. Endpoint-Network Correlation Workflow

As shown in Fig. 3, the correlation process does not directly convert every endpointnetwork pair into a link. Each pair must first pass the scoring and reasoning checks. This helps the framework avoid forced relationships when the available evidence is weak or one-sided.

Table 7. Endpoint-Network Correlation Output

Output	Meaning
Hard link	Strong endpoint-network relationship supported by correlation score and explicit reason
Soft link	Weaker relationship that remains useful as investigative context
No link	Pair does not have enough support to be retained
Network-only context	Network evidence is retained without forcing an endpoint relation
Empty correlation	Candidate evidence is insufficient to form cross-source context
Linked incident context	Context containing endpoint evidence, network evidence, links, scores, time gap, and relationship reasons

This output structure is useful because not every scenario produces balanced endpoint and network evidence. In a network-dominant case, for example, preserving a network-only context is more appropriate than forcing a weak endpoint-network link.

Algorithm 2. Endpoint-Network Correlation

Input: Endpoint candidate set E_c, network candidate set N_c, candidate evidence fields, timestamps, entity attributes, linker artifact, correlation weights (w_t, w_a, w_r), hard-link threshold τ_h, soft-link threshold τ_s, and correlation windows.
Output: Hard links, soft links, no-link records, linked incident context, network-only context, or empty correlation.

1. Read endpoint candidates E_c and network candidates N_c from records with to_correlate = True.
2. If one candidate side is empty, do not generate additional candidates automatically.
3. If network evidence is available and dominant, form a network-only context when appropriate.
4. Build compact text representations for endpoint candidates.
5. Build compact text representations for network candidates.
6. Transform endpoint and network texts using hashing, SVD, and normalization from the linker artifact.
7. For each endpoint-network pair, compute temporal distance $\Delta t(e_i, n_j)$.
8. Compute temporal proximity $T(e_i, n_j)$.
9. Compute entity consistency $A(e_i, n_j)$.
10. Compute behavioral relevance $R(e_i, n_j)$using cosine similarity.
11. Combine the three dimensions into correlation score $C(e_i, n_j)$.
12. Generate the relationship reason why from the supporting indicators.
13. Test the hard-link condition first.
14. If the pair satisfies the hard-link condition, store it as a hard link.
15. If the pair does not satisfy hard-link criteria, test the soft-link condition.
16. Store retained weaker pairs as soft links.
17. Ignore pairs that do not reach the soft-link threshold.
18. Export linked context, network-only context, or empty correlation according to the available evidence.

With this design, endpoint-network correlation serves as the incident-context construction stage. It does not validate the framework by itself and does not claim that every retained pair is certainly causal. Instead, it organizes retained evidence into stronger links, weaker links, or unsupported pairs. This distinction is important for the revised scope of the paper, because the framework is positioned as an incident-oriented correlation method rather than a ransomwarespecific detector.

3.7 Operational Parameters and Sensitivity Setup

The framework uses several operational thresholds to control candidate retention, MLsupported suspiciousness, and endpoint-network link formation. These values are not treated as universal security constants. They are fixed before the final evaluation and are used only as controlled settings for the evaluated scenarios. This is important because threshold-based and anomaly-based security analysis often involves trade-offs between sensitivity, noise, and false-positive exposure [15], [19], [16].

The most influential parameters are summarized in Table 8. Detailed implementation values such as full regular expressions, debugging options, and minor service-specific checks are not repeated in this section, because the main purpose here is to show which settings affect candidate selection and correlation behavior.

Table 8. Locked Operational Parameter Group

Group	Main Setting	Purpose
Endpoint rule gate	(p_H \geq 0.70)	Controls endpoint candidate retention from explicit rule evidence
Endpoint ML gate	rank (\geq 0.95), raw floor (\geq 0.05)	Controls active endpoint ML-supported candidates
Network ML gate	0.60 default; 0.95 for recon-aware setting	Controls active network ML-supported candidates
Scan-like threshold	25 unique ports	Identifies reconnaissance-like flow behavior
Correlation link gate	hard = 0.60; soft = 0.30	Separates stronger and weaker endpoint-network relations
Correlation time window	hard = 1 minute; soft = 10 minutes	Controls temporal support in link formation
Correlation safety controls	hard link requires reason; hard/soft exclusive; empty fallback disabled	Avoids duplicated links, unsupported hard links, and forced correlation

These parameters are locked before the final evaluation. The SAPPAN ground truth is not used to tune rule scores, ML scores, or correlation links during runtime [13], [14]. The values are therefore reported as part of the experimental configuration, not as optimized thresholds for all environments.

To address threshold robustness, the evaluation includes a sensitivity setup. The goal is not to search for the best threshold after seeing the results, but to observe whether candidate counts, link counts, and context type change sharply under stricter or more relaxed settings.

Table 9. Sensitivity Setup

Setup	Direction	Purpose
Default	Uses the locked values in Table 8	Main evaluation configuration
Conservative	Raises selected ML or correlation thresholds	Observes whether retained candidates and links remain under stricter filtering
Relaxed	Lowers selected thresholds or widens selected time windows	Observes whether additional candidates create meaningful or weak relations
Recon-aware	Applies stricter network ML and soft-link control when scan-like flow behavior is detected	Prevents network-heavy reconnaissance evidence from forcing unsupported endpoint-network links

The recon-aware setting is derived from flow behavior, such as repeated probing or many unique destination ports, not from the scenario name. This distinction is important because the framework should not depend on knowing the dataset label in advance. If endpoint evidence is insufficient while network evidence remains meaningful, the framework can preserve a network-only context instead of forcing a weak endpoint-network link.

3.8 Evaluation Strategy

The evaluation is conducted at the scenario level because the SAPPAN dataset provides annotated scenario evidence, but does not provide complete malicious or benign labels for every endpoint event, network flow, or endpoint-network pair [13], [14]. For

that reason, this study does not use event-level accuracy, precision, recall, F1-score, or ROC-AUC as the main evaluation metrics. Using those metrics without complete event-level labels could give a misleading impression of measurement precision.

Instead, the evaluation observes whether the framework output is consistent with the expected evidence and activity context of each scenario. The main outputs examined are retained endpoint candidates, retained network candidates, hard links, soft links, network-only context, and empty correlation. This is aligned with the purpose of the framework, which is to construct incident-oriented context from endpoint and network evidence, rather than to classify each individual record as malicious or benign.

The evaluation uses the scenario roles defined in Section 3.2. The Drupal vulnerability scenario-successful is used as the validation/development scenario, while Samba file-sharing known-creds and Samba file-sharing reconnaissance are used as final evaluation scenarios. The final scenarios are not used to train the ML components, tune rule scores, or adjust correlation

Table 10. Evaluation Focus

Evaluation Aspect	What is Observed	Purpose
Candidate retention	Number and proportion of retained endpoint and network candidates	Checks how much evidence is preserved before correlation
Rule and ML contribution	Rule-only, ML-only, and hybrid candidates	Shows the contribution of each scoring layer
Correlation output	Hard links, soft links, no links, and context type	Examines how retained candidates are connected
Scenario alignment	Match between framework output and SAPPAN expected evidence	Checks whether the output is consistent with scenario-level evidence
Sensitivity observation	Default, conservative, relaxed, and recon-aware settings	Observes whether results depend too heavily on one threshold setting

To address the role of each component, the evaluation also includes a simple ablation view. Rule-only output shows what is retained by explicit heuristic evidence. ML-only output shows candidates retained only through suspiciousness support. The hybrid output shows the combined candidate set used for correlation. This comparison is not used to claim that one layer is universally better than another; it is used to show how each layer affects the evidence passed into correlation.

In this study, external systems such as DEEPCASE and HOLMES are not reimplemented as experimental baselines because their input assumptions, event representation, training setup, and evaluation objectives are not directly equivalent to the proposed endpoint-network correlation pipeline. Applying them without reproducing their original data model and configuration could lead to an unfair or misleading comparison. Therefore, the comparative evaluation is limited to two forms: an internal ablation view and scenario-level alignment with SAPPAN evidence. The ablation view compares rule-only, ML-only, and hybrid candidate preservation, while the scenario-level alignment checks whether the retained candidates and correlation links are consistent with the expected evidence documented in the SAPPAN scenarios [10], [11], [13], [14].

The sensitivity observation is used to respond to concerns about threshold robustness. The framework is evaluated under the default configuration and then compared with stricter or more relaxed settings. The comparison focuses on candidate counts,

hard-link and soft-link counts, and final context type. In reconnaissance-heavy conditions, special attention is given to whether the framework preserves network evidence without forcing unsupported endpointnetwork links. This is important because event-correlation methods should consider context and relationships between evidence, not only isolated scores [8], [9], [12].

Overall, the evaluation is designed to be transparent about the available ground truth. The study does not claim universal threshold optimality or full detection performance across all enterprise environments. It evaluates whether the proposed framework can preserve relevant candidates, separate stronger and weaker relationships, and construct scenario-consistent incident context within the available SAPPAN evidence.

4. Results and Discussion

4.1 Evaluation Overview and Scenario-Level Ground-Truth Reference

The evaluation uses three SAPPAN scenarios: Drupal vulnerability scenario-successful, Samba file-sharing known-creds, and Samba file-sharing reconnaissance [13], [14]. Each scenario provides documented activity context and expected evidence, such as command execution, payload delivery, SMB interaction, C2-like communication, and reconnaissance behavior. In this study, this information is used as a scenario-level reference, not as complete event-level labels for every endpoint event, network flow, or endpoint-network pair.

For that reason, the evaluation does not treat the framework as a binary classifier and does not report event-level accuracy, precision, recall, F1-score, or ROC-AUC. Instead, the results are examined by checking whether the retained candidates, correlation links, and final context type are consistent with the expected scenario evidence. This is aligned with the purpose of the framework, which is to construct incident-oriented endpoint-network context rather than classify every individual record as malicious or benign.

Table 11. Scenario-level Expected Evidence and Evaluation Focus

Scenario	Expected Evidence / Context	Expected Context Type	Evaluation Focus
Drupal vulnerability scenario-successful	Drupal RCE, command execution, finger payload retrieval, PowerShell redirection, certutil decoding, payload execution, C2-like traffic	Linked endpoint-network context	Whether endpoint command evidence is related to network delivery and C2-like communication
Samba file-sharing known-creds	SMB interaction, known-credential exploitation, payload download, PowerShell execution, reverse-shell or C2-related traffic	Linked endpoint-network context	Whether SMB/service evidence is connected with endpoint-side execution behavior
Samba file-sharing reconnaissance	Port scan, SMB vulnerability probing, service discovery, many scan-like flows	Network-only or network-dominant context	Whether reconnaissance evidence is retained without forcing unsupported endpoint-network links

The later sections report the evaluation in four parts: candidate retention, rule-only/MLonly/hybrid contribution, endpoint-network correlation, and scenario-level alignment. This structure keeps the evaluation transparent while avoiding unsupported claims about full detection performance beyond the available SAPPAN evidence.

4.2 Candidate Retention Results

Candidate retention is the first result layer observed before endpoint-network correlation. At this stage, the framework does not correlate all raw endpoint events and network flows directly. Instead, it preserves records that contain rule-based evidence, ML-supported suspiciousness, or both. This is important because correlation is more meaningful when it is applied to selected evidence rather than to the full raw telemetry. In this study, the retained endpoint candidates E_c and network candidates N_c become the input for the correlation stage, where temporal, entity, and behavioral relationships are examined [8], [9], [12].

The candidate rate is calculated using the following equation:

$$\textit{Candidate Rate} = \frac{\text{Retained Candidates}}{\text{Total Records}} \times 100\% \tag{52}$$

For example, in the Drupal vulnerability scenario, the endpoint analysis retains 14 candidates from 118 endpoint events. The candidate rate is:

$$\frac{14}{118} \times 100\% = 11.86\%$$

This value only describes the proportion of records preserved for further analysis. It is not treated as detection accuracy, because the SAPPAN dataset provides scenario-level activity descriptions and expected evidence rather than complete malicious or benign labels for every event, flow, or endpoint-network pair [13], [14].

4.2.1 Endpoint Candidate Retention

The endpoint-side retention results are shown in Table 12. In the Drupal vulnerability scenario, 14 of 118 endpoint events are retained. Most of the retained records come from explicit command-related evidence, including finger activity, staging behavior, and process execution. The ML-supported candidates complement this result by preserving certutil decoding events, which are consistent with the documented payload decoding activity in the Drupal scenario [13].

In the Samba file-sharing known-creds scenario, four of 41 endpoint events are retained. These candidates are mainly related to PowerShell-based execution behavior, which fits the expected scenario context involving payload download and in-memory execution. In contrast, the Samba reconnaissance scenario does not produce endpoint candidates. This result is reasonable because the scenario is mainly characterized by network-side probing and port-scan behavior, not by strong endpoint-side execution traces.

Table 12. Endpoint Candidate Retention Results

Scenario	Total Endpoint Events	Rule-Based Candidates	ML-Supported Candidates	Retained Endpoint Candidates	Candidate Rate
Drupal vulnerability scenario-successful	118	12	2	14	11.86%
Samba file-sharing known-creds	41	4	0	4	9.76%
Samba file-sharing reconnaissance	65	0	0	0	0.00%

These endpoint results show that the framework does not force candidate retention when the endpoint evidence is weak. This is important for the later correlation stage, because unsupported endpoint candidates could create misleading endpoint-network links.

4.2.2 Network Candidate Retention

The network-side retention results are shown in Table 13. In the Drupal vulnerability scenario, three of ten network flows are retained. Two flows are retained through rule-based evidence related to port 79 service traffic, while one additional ML-supported flow appears on port 443. This is consistent with the scenario description, where payload delivery and C2-like communication are part of the expected activity context [13].

In the Samba file-sharing known-creds scenario, three of six network flows are retained. The rule layer captures SMB-related communication, while the ML layer preserves one HTTP flow. This result is useful because the scenario is not limited to SMB interaction alone; the documented activity also includes payload delivery through the supporting server [14]. Therefore, the retained network candidates provide service-level context that can later be compared with endpoint-side evidence during correlation.

Table 13. Network Candidate Retention Results

Scenario	Total Network Flows	Rule-Based Candidates	ML-Supported Candidates	Retained Network Candidates	Candidate Rate
Drupal vulnerability scenario-successful	10	2	1	3	30.00%
Samba file-sharing known-creds	6	2	1	3	50.00%
Samba file-sharing reconnaissance	2012	2011	0	2011	99.95%

The Samba reconnaissance scenario has a different pattern from the other two scenarios. A total of 2011 out of 2012 network flows were retained because the flow behavior forms a broad port-sweep pattern. This result should not be interpreted as highly selective filtering. In this case, the network layer works as a context-preservation layer because the expected evidence is network-dominant. The full retained count is still reported for transparency, while the next stage can limit exported candidates to keep the correlation process manageable.

Overall, the retention results follow the evidence orientation of each scenario. Drupal and Samba known-creds produce both endpoint and network candidates, so they can be evaluated further through endpoint-network correlation. Samba reconnaissance produces network evidence without endpoint candidates, which supports a network-only interpretation rather than a forced endpoint-network relation. This distinction is important because the framework is intended to preserve relevant evidence and prepare it for correlation, not to make every scenario appear as the same type of linked incident.

4.3 Rule-Only, ML-Only, and Hybrid Contribution

This section examines the contribution of each scoring layer before the candidates are passed into endpoint-network correlation. This ablation is used as an internal

comparative view, since the study does not reimplement external systems with different input assumptions and evaluation objectives. The purpose is not to prove that one layer is universally better than another, but to show how rule-based evidence and ML-supported suspiciousness behave when they are observed separately and when they are combined. This is important because anomalybased security analysis can support candidate discovery, but it is not always stable enough to replace explicit security evidence in every scenario [15], [16].

Three views are compared in this section. The rule-only view shows candidates retained only by explicit rule evidence. The ML-only view shows candidates retained when the ML layer is allowed to analyze the scenario without depending on rule retention. The hybrid view represents the main pipeline used in this study, where a record is retained when it is supported by either rule-based evidence or ML-supported suspiciousness. The hybrid result is therefore the candidate set used for the later endpoint-network correlation stage.

The endpoint-side comparison is shown in Table 14. In the Drupal scenario, the rule-only layer retains 12 endpoint candidates, while the ML-only view retains two candidates. These ML-only candidates are related to certutil decoding activity, which is consistent with the expected payload decoding step in the SAPPAN Drupal scenario [13]. When both layers are combined, the hybrid pipeline retains 14 endpoint candidates. This shows that ML contributes additional endpoint evidence in Drupal without replacing the explicit command and staging evidence retained by the rule layer.

Table 14. Endpoint-Side Rule-Only, ML-Only, and Hybrid Contribution

Scenario	Total Endpoint Events	Rule-Only Candidates	ML-Only Candidates	Hybrid Candidates	Main Observation
Drupal vulnerability scenario-successful	118	12	2	14	ML adds certutil decoding candidates beyond rule-retained command evidence
Samba file-sharing known-creds	41	4	4	4	ML identifies PowerShell-related activity, but the same evidence is already retained by rules in the hybrid pipeline
Samba file-sharing reconnaissance	65	0	0	0	No endpoint evidence is forced in a network-dominant reconnaissance scenario

In Samba known-creds, the ML-only view also retains four endpoint candidates. These candidates are related to PowerShell activity, which fits the expected payload download and execution context of the scenario [14]. However, in the hybrid pipeline, these records are already retained by the rule layer. This result is useful because it shows that the ML layer recognizes relevant endpoint behavior, but it does not necessarily add new endpoint candidates when explicit rule evidence already captures the same activity. In Samba reconnaissance, neither rule-only nor ML-only retains endpoint candidates, which supports the interpretation that this scenario is mainly network-dominant.

The network-side comparison is shown in Table 15. In the Drupal scenario, the rule-only view retains two network flows related to port 79 service traffic. The ML-only view retains two flows, including traffic related to port 79 and port 443. In

the hybrid pipeline, three network candidates are retained. This result indicates that the ML layer adds one relevant network candidate beyond the rule-retained port 79 traffic. The additional port 443 flow is consistent with the scenario context, where delivery and C2-like communication are part of the expected evidence [13].

Table 15. Network-Side Rule-Only, ML-Only, and Hybrid Contribution

Scenario	Total Network Flows	Rule-Only Candidates	ML-Only Candidates	Hybrid Candidates	Main Observation
Drupal vulnerability scenario-successful	10	2	2	3	ML adds a port 443 flow beyond the rule-retained port 79 evidence
Samba file-sharing known-creds	6	2	3	3	ML adds HTTP communication beyond SMB rule evidence
Samba file-sharing reconnaissance	2012	2011	2	2011	Rule-based port-sweep evidence preserves the broad reconnaissance context, while ML-only captures only a small part of it

In Samba known-creds, the rule-only layer retains SMB-related flows, while the MLonly view also retains an HTTP flow. The hybrid result keeps three network candidates, combining SMB service evidence with HTTP communication that is relevant to the payload delivery context described in the scenario [14]. This supports the role of ML as an additional suspiciousness layer, especially for flows that may not match the main service-based rule.

The Samba reconnaissance scenario shows the clearest difference between rule-only and ML-only behavior. Rule-only retention preserves the full port-sweep context, while ML-only captures only two flows. This does not make the ML layer useless; rather, it shows its limitation in representing broad scan behavior by itself. For this type of scenario, explicit port-sweep evidence is still needed to preserve the network context. This result supports the design choice used in this paper: ML is used as a supporting layer for candidate preservation, while rulebased evidence remains important for explicit and interpretable security context.

Overall, the ablation results show that the hybrid pipeline provides a practical balance between explicit evidence and anomaly-supported evidence. In Drupal and Samba known-creds, ML helps retain relevant candidates that are consistent with the documented scenario behavior. In Samba reconnaissance, the rule layer is more important because the scenario is dominated by broad network probing. These findings support the use of hybrid candidate preservation before correlation, without claiming that ML alone can replace rule-based analysis or existing security detection systems.

4.4 Endpoint-Network Correlation Results

After candidate retention, the retained endpoint and network candidates are passed into the correlation stage. This stage is needed because endpoint events and network flows explain different parts of an incident. Endpoint records can show command execution, process activity, or staging behavior, while network flows show communication with services, delivery hosts, or external infrastructure. If these two sources are

read separately, the incident context may remain fragmented. For that reason, the framework correlates retained candidates through temporal proximity, entity consistency, and behavioral relevance, following the role of event correlation in connecting heterogeneous security evidence [8], [9], [12].

The correlation results are summarized in Table 16. The numbers in the hard-link and softlink columns represent pair-level links. This means that one network flow can be linked to more than one endpoint event when several endpoint records support the same activity sequence.

Table 16. Endpoint-Network Correlation Results

Scenario	Endpoint Candidates	Network Candidates Used in Correlation	Hard Links	Soft Links	Final Context Type	Main Interpretation
Drupal vulnerability scenario-successful	14	3	8	32	Linked endpoint-network context	Endpoint command, staging, and decoding evidence are connected with port 79 delivery traffic and port 443 communication
Samba file-sharing known-creds	4	3	8	4	Linked endpoint-network context	PowerShell-related endpoint evidence is connected with SMB service traffic and HTTP delivery context
Samba file-sharing reconnaissance	0	300 exported network flows	0	0	Network-only context	Reconnaissance evidence is preserved as network-only context without forced endpoint links

To make the interpretation clearer, Table 17 summarizes how the framework separates hard links, soft links, no-link pairs, and network-only context.

Table 17. Correlation Output Assignment Logic

Output Type	Main Condition	Typical Evidence Pattern	Interpretation
Hard link	Strong correlation score with explicit relationship reason	Close time gap, victim/entity consistency, and clear service or behavior relation, such as `finger` activity with port 79 or PowerShell-related activity with SMB traffic	Stronger endpoint-network relation
Soft link	Relevant relation exists, but support is weaker than hard link	Same victim or related behavior is present, but the time gap is wider or the relation is less direct	Useful investigative context, but not treated as a strong relation
No link	Pair does not have enough support	Weak temporal, entity, or behavioral relation	Not retained as correlated evidence
Network-only context	Network evidence exists, but endpoint candidates are unavailable or insufficient	Port sweep, service probing, or scan-like traffic without supporting endpoint evidence	Network evidence is preserved without forcing endpoint-network correlation

In the Drupal vulnerability scenario, the framework forms 8 hard links and 32 soft links. The hard links mainly connect endpoint-side finger activity with port 79 traffic involving the victim host, supported by victim consistency and the finger-to-port-79 relation. Some hard links also connect staging-related endpoint activity with port 443 communication when timing and victim context are strong enough. This follows the documented Drupal scenario, where the vulnerable server retrieves, prepares, and executes a payload before communicating with the delivery and C2 infrastructure

[13].

The larger number of soft links in Drupal appears because several endpoint events still share the same victim and behavioral context, but occur farther from the related network flows. These links are not treated as confirmed causal relations. They are kept as weaker investigative context to preserve the broader activity sequence.

In the Samba file-sharing known-creds scenario, the framework produces 8 hard links and 4 soft links. The hard links connect SMB-related flows with PowerShell-related endpoint records through close timing and victim consistency. The soft links are mainly related to HTTP communication from the delivery server, which is relevant to the PowerShell download context but less direct than the SMB links. This matches the expected scenario context involving SMB access, payload delivery, and PowerShell-based execution [14].

The Samba reconnaissance scenario produces no hard or soft endpoint-network links because no endpoint candidates are available. The result is therefore kept as network-only context, which is appropriate for a scenario dominated by scanning and service probing. This avoids forcing endpoint-network relations that are not supported by the available evidence.

Overall, the correlation results show that Drupal and Samba known-creds produce linked endpoint-network context, while Samba reconnaissance remains network-only. This supports the revised scope of the paper: the framework organizes retained evidence into stronger links, weaker links, or one-sided context, without presenting itself as a universal detector.

4.5 Scenario-Level Ground-Truth Alignment

The correlation output is compared with the scenario-level evidence described in the SAPPAN dataset. This comparison is used to check whether the retained candidates and final context type are consistent with the documented activity in each scenario. The ground truth is not used as a complete event-level label, so this section does not report false-positive rate, precision, recall, or F1-score. Instead, the comparison is interpreted as scenario-level alignment between expected evidence and framework output [13], [14].

In the Drupal scenario, the framework output follows the main attack sequence described in the scenario documentation. The endpoint side preserves command and staging evidence, while the network side preserves port 79 and port 443 communication. The hard links mainly connect close finger-related endpoint events with port 79 traffic, while the soft links keep later related activities as weaker context. This result is consistent with the expected payload retrieval, staging, and communication flow described by SAPPAN [13].

In the Samba known-creds scenario, the output also matches the expected evidence. The endpoint candidates show PowerShell-related activity, while the network candidates preserve SMB communication and HTTP delivery context. The correlation stage connects these two evidence sources, which is important because the scenario is not represented by one isolated record. It involves service access, payload delivery, and endpoint-side execution behavior that need to be read together [14].

The Samba reconnaissance scenario is different. The expected evidence is mainly

Table 18. Scenario-Level Alignment between Expected Evidence and Framework Output

Scenario	Expected Evidence / Context	Observed Framework Evidence	Alignment
Drupal vulnerability scenario-successful	Drupal RCE, finger payload retrieval, PowerShell-based redirection, payload staging, certutil decoding, payload execution, and C2-like communication [13]	Endpoint candidates include finger, staging, process execution, and certutil decoding. Network candidates include port 79 traffic and port 443 communication. Correlation produces linked endpoint-network context with hard and soft links.	Aligned at scenario level
Samba file-sharing known-creds	SMB service access, known-credential exploitation, payload delivery, PowerShell-based execution, and reverse-shell or C2-related communication [14]	Endpoint candidates contain PowerShell-related activity. Network candidates include SMB traffic and HTTP delivery-related traffic. Correlation links endpoint activity with SMB and HTTP network evidence.	Aligned at scenario level
Samba file-sharing reconnaissance	Port scan, SMB vulnerability probing, service discovery, and scan-like network behavior [14]	Network candidates preserve the port-sweep context. No endpoint candidates are retained, and the final output remains network-only without forced endpoint-network links.	Aligned with network-dominant context

network-based, so the absence of endpoint links is not treated as a failure. The framework keeps the result as network-only context because there are no endpoint candidates strong enough to support a cross-source relation. This is an important outcome: the framework preserves reconnaissance evidence without making the result look more complete than the available data supports.

Overall, the comparison shows that the framework output is consistent with the documented scenario evidence. Drupal and Samba known-creds produce linked endpointnetwork context, while Samba reconnaissance remains network-only. This does not prove universal detection performance, but it supports the intended use of the framework as an incident-oriented correlation method that organizes retained evidence according to the available scenario context.

4.6 Threshold and Link-Strength Observations

This section discusses how the correlation output should be read under the hard-link and soft-link separation used in the framework. The purpose is not to claim that the selected thresholds are optimal for every environment. In anomaly-based and threshold-based security analysis, threshold selection often affects the balance between retained evidence, noise, and analyst workload [15], [19], [16]. For this reason, the results are interpreted through link strength rather than a single all-or-nothing correlation output.

The framework separates stronger and weaker relations by using two output levels. Hard links represent pairs with stronger temporal, entity, and behavioral support. Soft links preserve weaker relations that may still help explain the broader activity sequence. This separation is useful because security-event correlation often needs to keep contextual evidence available without treating every relation as equally strong [8], [9], [12].

In the Drupal scenario, the difference between the hard-link view and the hard-plus-soft view is quite visible. The 8 hard links represent the stronger relations, mainly involving close finger and staging-related evidence. The additional soft links do not change the final interpretation of the scenario, but they provide broader context

Table 19. Hard-Link and Soft-Link Observations

Scenario	Hard-Link View	Hard + Soft View	Interpretation
Drupal vulnerability scenario-successful	8 hard links	40 total links	Hard links capture closer endpoint-network relations, while soft links preserve broader activity around payload retrieval, staging, and communication
Samba file-sharing known-creds	8 hard links	12 total links	Hard links capture close SMB-related relations, while soft links add HTTP delivery context
Samba file-sharing reconnaissance	0 hard links	0 total links	No endpoint-network relation is forced; the result remains network-only

around related endpoint events and network flows. This means the soft links are useful for reading the incident sequence, but they should not be interpreted with the same strength as hard links.

In Samba known-creds, the hard-link result is more compact. The 8 hard links mainly connect SMB-related network traffic with PowerShell-related endpoint activity. The 4 soft links add HTTP delivery context, which is relevant to the expected scenario evidence but less direct than the SMB relations. This supports the use of soft links as supporting context rather than final strong relations.

The reconnaissance scenario provides an important boundary case. Even though many network flows are retained as reconnaissance evidence, the framework does not create hard or soft endpoint-network links because no endpoint candidates are available. This shows the role of the network-only guard: when evidence is one-sided, the framework preserves the available network context instead of forcing a cross-source relation.

Table 20. Threshold-Oriented Reading of Correlation Output

Reading Mode	What Is Emphasized	Effect on Interpretation
Hard-link reading	Stronger endpoint-network relations only	More conservative view; useful when only close and well-supported relations should be discussed
Hard + soft reading	Stronger relations plus weaker investigative context	Broader view; useful for reconstructing the activity sequence
Network-only reading	Network evidence without endpoint support	Prevents unsupported endpoint-network links in network-dominant scenarios

Overall, the threshold observation shows that the main scenario interpretation does not depend on treating every link equally. Drupal and Samba known-creds still produce linked endpoint-network context when only hard links are considered. Soft links mainly enrich the surrounding incident story. Samba reconnaissance remains network-only, which is important because it prevents the framework from overstating the evidence. This result supports a cautious interpretation of the framework output: hard links are stronger evidence, soft links are supporting context, and network-only output is used when cross-source correlation is not supported by the available candidates.

4.7 Discussion and Limitations

The results show that the framework behaves differently across the three evaluated scenarios. Drupal vulnerability and Samba known-creds produce linked endpoint-network context because both endpoint and network evidence are available and related.

In these two scenarios, the correlation stage helps connect command activity, service traffic, payload delivery, and execution-related behavior into a more readable incident context. This supports the role of endpoint-network correlation as a way to reduce fragmented analysis across heterogeneous evidence sources [8], [9], [12].

The Samba reconnaissance scenario shows a different but important outcome. The framework does not force endpoint-network links when endpoint evidence is unavailable. Instead, the result remains as network-only context. This is useful because reconnaissance activity can be visible mainly through network behavior, especially when the available endpoint telemetry does not contain strong execution evidence. In this case, preserving the network context is more appropriate than creating weak endpoint-network relations that are not supported by the data.

The ablation results also clarify the role of rule-based and ML-supported analysis. Rulebased scoring is useful for retaining explicit and interpretable evidence, such as service traffic, command activity, and scan-like behavior. The ML layer helps preserve additional unusual records, such as certutil decoding in Drupal or HTTP delivery-related traffic in Samba knowncreds. However, the results also show that ML alone does not replace rule-based evidence. This is especially visible in the reconnaissance scenario, where broad port-sweep behavior is better preserved by explicit network rules than by ML-only scoring. This finding is consistent with the general challenge of anomaly-based security analysis, where unusualness does not always translate directly into complete incident understanding [15], [16].

The hard-link and soft-link separation is also useful for reading the correlation output. Hard links give a more conservative view of stronger endpoint-network relations, while soft links preserve weaker context that may still help reconstruct the activity sequence. This prevents all links from being interpreted with the same strength. For incident analysis, this distinction is important because some evidence pairs are closely supported by time, entity, and behavior, while others are only useful as broader investigative context.

Table 21. Main Findings and Limitations

Aspect	Main Finding	Limitation
Endpoint-network correlation	Drupal and Samba known-creds produce linked context	The result is based on a small number of public scenarios
Network-only handling	Samba reconnaissance is preserved without forced endpoint links	Network-dominant cases may retain many flows and require careful filtering
Rule and ML contribution	ML supports candidate preservation, but rules remain important for explicit evidence	ML-only output is not sufficient to replace rule-based security logic
Ground-truth comparison	Output is aligned with scenario-level expected evidence	Complete event-level labels are not available, so precision, recall, F1-score, and false-positive rate are not reported
Threshold use	Hard and soft links help separate stronger and weaker relations	Thresholds are operational settings and may need adjustment in other environments

Several limitations should be noted. First, the evaluation uses only three public SAPPAN scenarios [13], [14]. These scenarios are useful because they provide documented endpoint and network evidence, but they do not represent the full diversity of enterprise environments. Second, the ground truth is scenario-level rather than event-level. For that reason, this study does not claim event-level detection performance or report supervised classification metrics. Third, the reconnaissance scenario

produces broad network retention, which shows that the framework can preserve scan context but also highlights a selectivity limitation in networkheavy cases.

Finally, the framework should be interpreted within its intended scope: supporting incident-oriented correlation from retained endpoint and network evidence. Its contribution is in organizing evidence into stronger links, weaker links, or one-sided context. Further work should evaluate the approach on more diverse scenarios, larger enterprise telemetry, and richer ground-truth annotations so that event-level performance and operational false-positive behavior can be measured more directly.

5. Conclusion

This study presented an incident-oriented endpoint-network correlation framework for organizing security evidence from host events and network flows. The framework combines rule-based scoring, ML-supported suspiciousness, hybrid candidate preservation, and endpoint-network correlation to construct incident context from selected evidence. Rather than treating each record as a final malicious or benign decision, the framework focuses on preserving relevant candidates and examining how they relate across evidence sources.

The evaluation shows different behavior across the three SAPPAN scenarios. In the Drupal vulnerability scenario, the framework retained 14 endpoint candidates and 3 network candidates, then produced 8 hard links and 32 soft links. These links connected endpoint-side command, staging, and decoding evidence with port 79 delivery traffic and port 443 communication. In the Samba file-sharing known-creds scenario, the framework retained 4 endpoint candidates and 3 network candidates, resulting in 8 hard links and 4 soft links. The linked context connected PowerShell-related endpoint activity with SMB and HTTP network evidence. In the Samba file-sharing reconnaissance scenario, no endpoint candidates were retained, while 2011 network flows were preserved as reconnaissance context. The final output remained network-only, without forcing endpoint-network links.

The results also show the role of each analysis layer. Rule-based scoring remains important for explicit and interpretable evidence, such as command activity, service traffic, and portsweep behavior. The ML layer helps preserve additional suspicious candidates, such as decoding activity and delivery-related traffic, but it is better treated as a supporting layer rather than a replacement for rule-based analysis. This is especially clear in the reconnaissance scenario, where broad scan behavior is better represented by explicit network evidence than by ML-only scoring.

The hard-link and soft-link separation also helps make the correlation output easier to interpret. Hard links provide a stronger view of endpoint-network relationships, while soft links preserve weaker context that may still support the incident story. This distinction is useful because not every related event pair should be read with the same strength. In networkdominant cases, the framework can also preserve network-only context when endpoint evidence is not available.

There are still several limitations. The evaluation is based on three public scenarios, so the results should not be generalized to all enterprise environments. The available ground truth is scenario-level rather than complete event-level labeling, so this study does not report precision, recall, F1-score, or operational false-positive rate. Future

work should evaluate the framework using larger and more diverse datasets, richer event-level or pair-level annotations, and more realistic operational telemetry. Improvements are also needed for handling networkheavy reconnaissance cases so that broad scan context can be preserved without producing excessive candidate volume.

References

[1] Rick Hofstede et al. "Flow Monitoring Explained: From Packet Capture to Data Analysis with NetFlow and IPFIX". In: *IEEE Communications Surveys & Tutorials* 16.4 (2014), pp. 2037–2064. DOI: 10.1109/COMST.2014.2321898.

[2] Min Du et al. "DeepLog: Anomaly Detection and Diagnosis from System Logs through Deep Learning". In: *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 2017, pp. 1285–1298. DOI: 10.1145/3133956.3134015.

[3] Tomáš Jirsík and Pavel Velan. "Host Behavior in Computer Network: One-Year Study". In: *IEEE Transactions on Network and Service Management* 18.1 (2021), pp. 822–838. DOI: 10.1109/TNSM.2020.3041493.

[4] Markus Ring et al. "A Survey of Network-Based Intrusion Detection Data Sets". In: *Computers & Security* 86 (2019), pp. 147–167. DOI: 10.1016/j.cose.2019.06.005.

[5] Amin Kharraz et al. "UNVEIL: A Large-Scale, Automated Approach to Detecting Ransomware". In: *Proceedings of the 25th USENIX Security Symposium (USENIX Security '16)*. 2016, pp. 757–772.

[6] Daniele Sgandurra et al. "Automated Dynamic Analysis of Ransomware: Benefits, Limitations and Use for Detection". In: *arXiv preprint* (2016). arXiv: 1609.03020 [cs.CR].

[7] H. Alraizza and A. Algarni. "Ransomware Detection Using Machine Learning: A Survey". In: *Big Data and Cognitive Computing* 7.3 (2023), p. 143. DOI: 10.3390/bdcc7030143.

[8] Evangelos Raftopoulos, Martin Egli, and Xenofontas Dimitropoulos. "Shedding Light on Log Correlation in Network Forensics Analysis". In: *Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA)*. 2012, pp. 232–241. DOI: 10.1007/978-3-642-31427-7_12.

[9] Stanislav Špaček and Pavel Čeleda. "Threat Detection Through Correlation of Network Flows and Logs". In: *Proceedings of the International Conference on Autonomous Infrastructure, Management and Security (AIMS)*. 2018. DOI: 10.1007/978-3-030-01701-9_11.

[10] T. van Ede et al. "DEEPCASE: Semi-Supervised Contextual Analysis of Security Events". In: *Proceedings of the IEEE Symposium on Security and Privacy (SP)*. 2022. DOI: 10.1109/SP46214.2022.9833640.

[11] S. M. Milajerdi et al. "HOLMES: Real-Time APT Detection Through Correlation of Suspicious Information Flows". In: *arXiv preprint* (2018). arXiv: 1810.01594 [cs.CR].

[12] D. Levshun and I. Kotenko. "A Survey on Artificial Intelligence Techniques for Security Event Correlation: Models, Challenges, and Opportunities". In: *Artificial Intelligence Review* 56 (2023), pp. 8547–8590. DOI: 10.1007/s10462-022-10331-8.

[13] Martin Cermák and Tomáš Jirsík. *D3.2 Annotated Dataset*. SAPPAN Consortium Deliverable. 2020.

[14] Martin Cermák et al. "Towards Provable Network Traffic Measurement and Analysis via Semi-Labeled Trace Datasets". In: *Proceedings of the Traffic Measurement and Analysis Conference (TMA)*. 2018. DOI: 10.23919/TMA.2018.8506503.

[15] Sami Khraisat et al. "Survey of Intrusion Detection Systems: Techniques, Datasets and Challenges". In: *Cybersecurity* 2.1 (2019), pp. 1–22. DOI: 10.1186/s42400-019-0038-7.

[16] Z. Yang et al. "A Systematic Literature Review of Methods and Datasets for Anomaly-Based Network Intrusion Detection". In: *Computers & Security* 116 (2022), p. 102675. DOI: 10.1016/j.cose.2022.102675.

[17] Nicola Capuano et al. "Explainable Artificial Intelligence in Cybersecurity: A Survey". In: *IEEE Access* 10 (2022), pp. 93575–93600. DOI: 10.1109/ACCESS.2022.3200173.

[18] I. H. Sarker et al. "Explainable AI for Cybersecurity Automation, Intelligence and Trustworthiness in Digital Twin: Methods, Taxonomy, Challenges and Prospects". In: *ICT Express* (2024). DOI: 10.1016/j.icte.2024.03.007.

[19] Ali Ghafouri et al. "Optimal Thresholds for Anomaly-Based Intrusion Detection in Dynamical Environments". In: *Decision and Game Theory for Security*. Springer, 2016, pp. 415–434. DOI: 10.1007/978-3-319-47413-7_21.