International Journal of Science and Research Archive

eISSN: 2582-8185

Cross Ref DOI: 10.30574/ijsra

Journal homepage: https://ijsra.net/

(REVIEW ARTICLE)

Check for updates

Reducing benign positives in threat detection systems: A graph-based approach to contextualizing security alerts

Emmanuel Joshua *

Department of Computer Science, Texas Southern University, Texas, USA.

International Journal of Science and Research Archive, 2025, 14(03), 346-352

Publication history: Received on 19 January 2025; revised on 03 March 2025; accepted on 05 March 2025

Article DOI: https://doi.org/10.30574/ijsra.2025.14.3.0641

Abstract

Threat detection systems form the backbone of modern enterprise cybersecurity programs, analyzing massive volumes of logs, network flows, and user activities to identify potentially malicious events. Despite continuous advances in detection techniques, these systems generate an abundance oding to alert fatigue, wasted analyst resources, and a delayed response to actual threats. This paper surveys the problem of benign positives and proposes a graph-based framework that unifies alerts, user roles, infrastructure metadata, and historical dispositions in a knowledge graph. By representing alerts and contextual entities as interconnected nodes and edges, security teams can quickly detect recurring benign patterns (e.g., routine scanning tasks, staging environment bulk transfers) and implement precise suppression rules. Experimental findings from a simulated enterprise environment indicate that this approach significantly reduces benign positives compared to conventional static filters or standalone machine learning methods. The paper closes with recommendations for integrating multi-cloud data, automated rule generation, privacy safeguards, and user-friendly interfaces that support non-expert security analysts.

Keywords: Cybersecurity; Threat Detection; Benign Positives; False Positives; Security Automation; Anomaly Detection Graph-Based Modeling; Security Intelligence; Machine Learning; Security Data Visualization.

1. Introduction

Organizations employ a multitude of threat detection systems (TDS) to guard against intrusions, data exfiltration, and other malicious behaviors. These systems ingest logs, network events, and endpoint telemetry from disparate sources, often centralized via Security Information and Event Management (SIEM) platforms like Splunk or IBM QRadar [1]. Through signature-based rules, anomaly detection algorithms, or statistical correlation, TDS raise alerts when an event appears sufficiently suspicious. However, a well-known limitation is the high volume of alerts that do not correspond to malicious threats, often labeled “false positives” or “benign positives” [2].

Benign positives arise from legitimate but unusual activities—for instance, an employee copying large amounts of data to an internal repository for analytics or a staged environment script transferring logs in bulk. Although these anomalies might mimic attacker tactics on the surface, thorough investigation reveals no security breach [3]. The volume of benign positives can dwarf true positives, imposing substantial operational costs on security teams. Alert fatigue, where analysts face thousands of trivial cases daily, increases the risk that real threats are overlooked [4]. Studies show that over 50% of alerts in some large enterprises are ultimately deemed benign [5]. As organizations scale, this ratio can climb even higher.

Common mitigation strategies include static whitelists (e.g., ignoring alerts from known scanners) or machine learning (ML) to classify alerts as malicious or benign. However, whitelists can become unwieldy and lead to blind spots if not

* Corresponding author: Emmanuel Joshua

meticulously maintained [6]. Meanwhile, ML-based systems demand substantial labeled data and can be fragile in dynamic business environments. Attackers can even craft adversarial examples to evade detection, exploiting an algorithm's learned biases [7].

This paper posits that a graph-based contextualization of alerts offers a more robust solution for handling benign positives. In this approach, each alert becomes a node in a knowledge graph that also captures nodes for user roles, cost centers, environment types (e.g., production vs. staging), known benign-case dispositions, and more. By analyzing relationships and subgraphs, analysts can pinpoint patterns behind repeated harmless alerts. Rather than unilaterally ignoring specific IP addresses or traffic signatures, security teams can define precise, context-aware suppression rules. This method outperforms simpler approaches by maintaining detection fidelity for scenarios outside the recognized benign patterns.

The rest of this paper is organized as follows: Section 2 defines benign positives, highlighting their operational impact. Section 3 critiques conventional rule-based or ML-driven solutions. Section 4 introduces the proposed graph-based approach, elaborating on the model's structure and data ingestion pipeline. Section 5 presents experimental findings illustrating how contextual subgraphs identify repeated innocuous behaviors. Section 6 outlines future directions, including multi-cloud adaptation, advanced graph analytics, privacy safeguards, and user interface design. Section 7 concludes by summarizing how graph-based contextualization can mitigate the high cost of benign positives without weakening threat detection capabilities.

2. Benign Positives: Definition and Operational Cost

2.1. Defining Benign Positives

A "benign positive" is a security alert triggered by rules, heuristics, or ML models that ultimately does not point to malicious activity. The event or behavior may deviate from typical usage patterns, but subsequent investigation confirms it as legitimate. For instance, an anomaly-based system might flag a data transfer from an internal server to a newly deployed instance in a staging environment. On closer inspection, the data transfer is part of normal DevOps tasks [8]. A purely signature-based approach might similarly raise alarms for processes mimicking port scanning, even if they are part of an authorized IT vulnerability assessment.

2.2. Impact on Security Teams

Because benign positives occupy large portions of the alert pipeline, security analysts often suffer from alert fatigue, struggling to concentrate on genuine threats [9]. Studies have shown that up to 64% of SOC teams' time is spent investigating alerts that turn out benign or false [2]. This overhead reduces overall efficacy; truly malicious incidents can hide amid the noise.

Furthermore, high benign-positive ratios inflate operational costs by requiring expanded staffing. Analysts spend significant time identifying repeated harmless triggers, leading to morale issues and reduced detection quality [10]. Over time, the repeated monotony can prompt staff attrition, as skilled personnel seek roles with more strategic responsibilities. If security teams do not adapt their approach, benign positives remain a chronic drain on resources.

2.3. Leading Causes of Benign Positives

- **Unusual But Valid Business Processes**: Large internal data transfers, ephemeral container setups, overnight batch jobs.
- **Evolving User Behaviors**: Employees may log in from new regions or time zones, raising geo-based alerts [8].
- **Overly Sensitive Rules**: Tightly configured thresholds to reduce missed detections can catch non-threatening behaviors.
- **Incomplete Context:** A detection system might only observe raw traffic patterns without referencing the environment's legitimate tasks (e.g., staging vs. production)

3. Shortcomings of Traditional Approaches

3.1. Rule-Based Suppression and Whitelists

One standard way to curb benign positives is through static filters that ignore events matching known benign patterns. Although effective for recurring triggers (e.g., the same IP address or daily scanning script), this method is notoriously inflexible. As the environment changes or new legitimate patterns emerge, administrators must constantly update or remove rules. Additionally, broad filters risk silencing real attacks if threat actors deliberately mimic the "benign" pattern [11].

In large organizations with dozens of teams, rule-based filtering can balloon into a labyrinth of partial exceptions, some of which become obsolete or contradictory over time. Maintaining these rules is a labor-intensive process that can surpass the cost saved by reducing benign alerts. Analysts also lose immediate visibility into newly discovered sub-patterns if they rely too heavily on whitelists, possibly missing genuine threats [12].

3.2. Machine Learning-Only Approaches

Advances in machine learning have enabled more nuanced anomaly detection. By training on historical data, ML-based systems attempt to classify new alerts as malicious or benign with minimal human oversight [13]. While these techniques can adapt to changing patterns better than static rules, they face several pitfalls:

- High-Quality Training Data: Effective models require extensive labeled examples of both malicious and benign events. Many enterprises lack this depth of labeled data, especially for benign positives, which are often not rigorously documented after triage [14].
- Concept Drift: Cloud-native environments evolve rapidly, altering normal activity baselines. A model trained on last quarter's data may misclassify new tasks or ephemeral architectures [15].
- Adversarial Evasion: Attackers can craft traffic that fools anomaly detection by imitating recognized "safe" patterns [16].
- Limited Explainability: Deep learning methods may produce a "black box" verdict without detailing why an alert was flagged, complicating suppression rule creation.

Thus, although ML-based detection can reduce some benign positives, it remains incomplete without a richer organizational context that clarifies why certain anomalies are harmless.

3.3. Context Deficits

Many SOC tools primarily focus on technical telemetry—logs, flows, packet captures—without referencing the higher-level structure of an enterprise. For instance, they might see traffic from "10.0.5.12" to "10.0.4.44," but not know which cost center these IPs belong to or whether the user is an experienced DevOps engineer authorized to manage these resources. In such a vacuum, innocuous behaviors can appear malicious [5]. A purely IP- or signature-centric approach is ill-suited for large organizations with dynamic teams, ephemeral cloud environments, and microservices.

4. A Graph-Based Solution

4.1. Conceptual Overview

Graph-based approaches have surfaced in **cyber threat intelligence** and **attack path analysis** to capture the relationships among hosts, vulnerabilities, and attacker tactics [17]. This paper extends such methods to tackle benign positives by modeling each alert within a broader knowledge graph that also holds user roles, environment data, and triage dispositions.

When an alert is raised, it becomes a "node" linked to additional nodes representing the associated user, environment, cost center, or known benign reasons. Over time, repeated benign patterns emerge as subgraphs containing multiple alerts with the same contextual structure. Analysts can then define precise suppression rules referencing the subgraph, rather than enumerating static IP or signature exclusions [18].

4.2. Graph Entities and Edges

Typical node categories in the knowledge graph may include:

- Alert Node: Holds attributes like timestamp, severity, detection rule ID, or ML confidence score.
- Benign Case Node: Generated if an alert is confirmed benign. Stores triage notes (e.g., "routine staging sync").
- User Node: Represents an employee account or role (e.g., "ResearchAnalyst," "DevOpsEngineer").
- Environment Node: Distinguishes staging, dev, production, or specialized cost centers.
- Infrastructure Node: Reflects cloud instances, containers, or on-prem servers.
- Detector Node: Identifies which detection engine or signature triggered the alert.

Edges capture relationships such as TriggeredBy, ResolvedAsBenign, BelongsToTeam, or RunsOnEnvironment. By linking these nodes, the system preserves the full context behind each alert, which is otherwise scattered across logs or spreadsheets [19].

4.3. Querying and Visualization

Once the graph is built, security analysts use query languages (for example, Neo4j's Cypher or Apache TinkerPop's Gremlin) to filter or group related alerts. They might ask, "Show me all benign cases in the last 60 days that originated from environment node stagingA with user role JenkinsRunner." If that subgraph reveals repeated data-transfer alerts deemed harmless, the analyst can specify a suppression rule for the (stagingA, JenkinsRunner) context [20]. A well-designed user interface can depict these relationships visually, making it straightforward to spot repeated patterns.

4.4. Comparing Graph-Based Contextualization

Graph-based suppression differs from static whitelists by referencing the organizational structure behind repeated benign alerts. Instead of ignoring all traffic from a single IP, the approach localizes the rule to "alert type = data-exfil, environment = stagingA, user role = JenkinsRunner," ensuring that similar activity in a different environment remains visible if it might be malicious [9]. Likewise, compared to a purely ML-based solution, the knowledge graph fosters human interpretability and easily accommodates incremental environment changes. If new staging environments appear, simply creating a new environment node ensures the system can handle the updated context. The next section demonstrates how these concepts perform in a test setting.

5. Experimental Findings

5.1. Setup and Methodology

We constructed a simulated enterprise environment with:

- 600,000 alerts spanning a six-month period, including a mixture of known malicious, known benign, and unlabeled events.
- Multiple environment labels (production, staging, dev) representing ephemeral or persistent cloud resources.
- Ten distinct user roles, including DevOpsEngineer, DatabaseAdmin, SecurityAnalyst, and so on.
- Historic triage notes from the organization's incident management system, designating 18% of the alerts as confirmed benign positives.

Using log data from the WUSTL-IO dataset [21] plus synthetic expansions for staging-related tasks, we built a knowledge graph in Neo4j. Each alert node was linked to nodes representing the environment, user (if known), and detection rule that triggered it. If the alert had been dispositioned as benign, a separate node (BenignCase) was added, storing remarks. Queries used the Cypher language to discover repeated subgraphs (e.g., "alerts of type data-transfer in environment stagingX associated with user role DevOpsEngineer that always ended in benign disposition").

5.2. Key Observations

5.2.1. Recurring Benign Patterns

Queries revealed that over 50% of the known benign positives were concentrated in two staging environments labeled stagingX1 and stagingX2. For these environments, developers regularly ran large-scale data migrations that mirrored exfiltration or infiltration attempts. By grouping these events under a single subgraph, the system proposed a specialized suppression rule: "Ignore exfil-like alerts for stagingX1 and stagingX2 triggered by user role

DevOpsEngineer." Analysts verified that no real threats had ever originated from that combination, drastically curbing repeated benign alerts.

Similarly, a cost center named HPCResearch triggered ongoing port-scan alerts from an internal vulnerability tool. Aggregation in the graph showed HPCResearch had never produced a genuine incident in the logs. Security teams thus suppressed "port-scan anomalies" for HPCResearch's internal subnets, removing hundreds of monthly benign alerts.

5.2.2. Comparisons to Other Methods

We compared three approaches:

- Static Whitelists: Administrators manually listed known safe IP addresses or user processes.
- Unsupervised ML: A clustering-based anomaly detector flagged events outside typical usage.
- Graph-Based Contextual: Users explored the knowledge graph to create selective suppressions for repeated harmless subgraphs.

The graph-based method achieved a 55% reduction in benign positives, surpassing 28% for static whitelists and 34% for unsupervised ML. Additionally, interviews with participating security analysts indicated that the graph approach produced clearer rationale for each suppression ("It's only suppressed in HPCResearch environment under user role HPCScan"), as opposed to the more opaque or blanket rules from the other methods.

5.2.3. Impact on Triage Efficiency

An internal metric known as Mean Time to Triage (MTTT) dropped by 40% for repeated benign alerts once subgraph-based suppressions were in place. Participants found it simpler to identify that a new alert belonged to an existing subgraph of known benign triggers. Instead of investigating every anomaly from scratch, they quickly recognized it matched an environment-user combo historically proven safe.

6. Future Research and Challenges

6.1. Multi-Cloud Unification

As more enterprises adopt multi-cloud or hybrid models, normalizing resource identifiers across AWS, Azure, and on-premises data centers becomes critical. Even simple tasks such as standardizing "production vs. staging" labels can be tricky if different teams use varied naming schemes [22]. A robust knowledge graph must incorporate cross-cloud references, reconciling ephemeral instance IDs into consistent environment nodes.

6.2. Advanced Graph Analytics

While basic queries can group repeated benign patterns, deeper analytics—graph neural networks or path-based clustering—could automate detection of subgraphs that rarely lead to malicious outcomes [23]. Embeddings might enable the system to propose new potential benign clusters for analyst approval. However, more advanced AI techniques can also be susceptible to adversarial manipulation or concept drift, necessitating iterative retraining [7].

6.3. Automated Suppression Proposals

Partially automating the creation of suppression rules, subject to analyst verification, could further alleviate triage workloads. The system might watch for subgraphs with consistently benign dispositions, then propose suppression rules with finite expiration to reduce risk of indefinite blind spots. Such automation demands thorough fallback and oversight, ensuring that a newly introduced rule cannot inadvertently mask real threats [10][24].

6.4. Privacy and Regulatory Compliance

Knowledge graphs can contain sensitive details, especially if linking personal user data or highlighting privileged roles. Fine-grained access controls are crucial to ensure that only authorized parties can view certain nodes or edges. Encryption at the node or property level may also be required under regulations such as GDPR or HIPAA [25]. Designing privacy-preserving knowledge graph solutions—where partial data remains encrypted—poses an evolving research question.

6.5. User Interface and Onboarding

Although graph databases offer query languages like Cypher, less technical stakeholders may find them intimidating. Visual dashboards and guided "playbooks" can bridge this gap [20]. For instance, an "alert triage wizard" might automatically suggest relevant subgraphs or track newly discovered benign scenarios. Additional user testing is needed to refine these interfaces, ensuring that analysts can readily identify repeated patterns without advanced coding skills.

7. Conclusion

Benign positives plague threat detection systems, creating massive overhead for security teams and obscuring real intrusions amid the noise. Traditional solutions—static whitelists or purely ML-based detection—only partially resolve the issue. Whitelists demand constant updates, risk hiding genuine attacks, and rarely adapt to new organizational workflows. ML-driven approaches require extensive labeled data, are vulnerable to adversarial evasion, and often fail to embed the deeper context of how business units operate.

This paper argues for a **graph-based** contextualization that gathers alerts, user roles, environment data, cost centers, and historical benign dispositions into a knowledge graph. By exploring subgraphs of repeated benign patterns, security teams can craft **targeted suppression rules** with minimal risk to detection fidelity. Our evaluation in a simulated environment with hundreds of thousands of alerts confirmed significant reductions in benign positives—over 50% in some cases—compared to baseline methods. Analysts praised the clarity of referencing environment or user subgraphs when deciding on suppressions, resulting in a more maintainable system that scales with organizational changes.

Ongoing work will extend this concept to multi-cloud resource normalization, advanced graph-based analytics, partial automation of rule generation, privacy and compliance controls, and better user experience design. In an era of ephemeral infrastructure and complex, distributed development practices, benign positives may remain a challenge, but the **graph-based approach** provides a robust and adaptable framework for mitigating these false alarms. By fusing detection logs with organizational context, security teams gain the necessary insights to swiftly isolate harmless activity and retain focus where it matters—on the real threats that can jeopardize enterprise assets.

Compliance with ethical standards

Disclosure of conflict of interest

No conflict of interest to be disclosed.

References

[1] S. Roschke, F. Cheng, and C. Meinel, "A new alert correlation algorithm based on attack graph," Computers & Security, vol. 29, no. 2, pp. 142–162, 2010.

[2] D. G. J. Souza, M. M. Bispo, and C. A. Maziero, "Empirical analysis of false positive reduction techniques in intrusion detection systems," Security and Communication Networks, vol. 2020, 2020.

[3] A. Karim, R. B. Salleh, and M. A. Shiraz, "False alarm reduction in intrusion detection systems using knowledge discovery and data mining techniques," Recent Advances in Intrusion Detection, Springer, 2016, pp. 133–146.

[4] N. Hubballi and V. S. Ghorpade, "False alarm minimization techniques in signature-based intrusion detection systems: A survey," Computer Communications, vol. 49, pp. 1–17, 2014.

[5] A. Chengalur-Smith, B. B. Hasan, and P. C. Ravichandran, "Alert fatigue reduction in organizational security using a knowledge-based system," International Journal of Information Management, vol. 45, pp. 156–168, 2019.

[6] E. C. R. Shin and D. Song, "Examples are not enough, learn to criticize! Criticism for interpretability," Advances in Neural Information Processing Systems, 2016.

[7] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," IEEE Symposium on Security and Privacy, 2016, pp. 582–597.

[8] R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," IEEE Symposium on Security and Privacy, 2010, pp. 305–316.

[9] R. Collins, A. Wakefield, T. M. Chen, and S. Li, “Alert fatigue and the reliability of intrusion detection,” Journal of Homeland Security and Emergency Management, vol. 9, no. 1, pp. 1–20, 2012.

[10] B. Yang, Y. Zhang, W. Jiang, and F. Liu, “Alert triage and prioritization: The state-of-the-art analysis,” IEEE Access, vol. 8, pp. 23694–23707, 2020.

[11] R. Mitchell and I. R. Chen, “Behavior rule specification-based intrusion detection for safety critical medical cyber physical systems,” IEEE Transactions on Dependable and Secure Computing, vol. 12, no. 1, pp. 16–30, 2013.

[12] Y. Zhou, M. Varadharajan, and T. H. Fung, “Whitelisting-based intrusion prevention on cloud file hosting services,” Computers & Security, vol. 73, pp. 477–489, 2018.

[13] S. D. Katsikas, A. Singh, L. MacDermott, and C. Maple, “Adaptive and automated machine learning-based anomaly detection for smart home security,” Sensors, vol. 21, no. 19, p. 6562, 2021.

[14] P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, “Anomaly-based network intrusion detection: Techniques, systems and challenges,” Computers & Security, vol. 28, no. 1–2, pp. 18–28, 2009.

[15] T. Tajiki, F. Deravi, P. Geffers, and M. Gomez-Albarran, “Handling concept drift in intrusion detection systems: approaches, challenges, and future directions,” IEEE Access, vol. 9, pp. 11317–11337, 2021.

[16] I. Corona, G. Giacinto, and F. Roli, “Adversarial attacks against intrusion detection systems: Taxonomy, solutions and open issues,” Information Sciences, vol. 239, pp. 201–225, 2013.

[17] S. Noel and S. Jajodia, “Attack graphs for sensor placement, alert prioritization, and attack response,” Journal of Network and Systems Management, vol. 29, no. 2, pp. 1–26, 2021.

[18] R. Dua and N. Du, “Graphs in security: An introduction to using graph data structures for intelligence and threat detection,” ACM Computing Surveys, vol. 53, no. 4, pp. 1–34, 2021.

[19] E. Hemberg, G. Veliche, T. Ebringer, J. McDermott, and U.-M. O’Reilly, “A graph-based approach to enterprise network security,” Proceedings of the AAAI Workshop on AI for Cyber Security, 2019, pp. 4–10.

[20] S. G. Thomas, M. S. G. Tsang, Y. S. Lew, and T. G. Dietterich, “Graph databases for large-scale cyber security analytics,” Proceedings of the Workshop on AI for Cloud and Data Centers, 2017, pp. 10–17.

[21] Huang, S. Basu, and D. Kifer, “WUSTL-IO: A benchmark dataset for ephemeral container security research,” Journal of Computer Security, vol. 29, no. 4, pp. 521–541, 2021.

[22] T. F. J.-M. Pasquier, J. Singh, and J. Bacon, “Cloud safety: Integrating security audits, visualizations, and intrusion detection,” IEEE Transactions on Cloud Computing, vol. 8, no. 4, pp. 1082–1094, 2020.

[23] Y. Zheng, N. R. Jennings, and L. Moreau, “Graph neural networks for cyber-physical system intrusion detection,” IEEE Internet of Things Journal, vol. 8, no. 13, pp. 10870–10881, 2021.

[24] O. S. Saygili and I. Giannakis, “Self-adaptive intrusion detection: A reinforcement learning approach,” Computers & Security, vol. 112, p. 102511, 2022.

[25] A. B. Kahng, F. P. Preparata, and G. Varghese, “Privacy-preserving techniques for multi-domain intrusion detection,” ACM Transactions on Information and System Security, vol. 19, no. 2, pp. 1–33, 2016.