

DOI: https://doi.org/10.48009/3_iis_2026_135

A Validation-Oriented Framework for Automated Cybersecurity Dataset Labeling

David Charles Loper, *University of North Dakota*, david.loper@und.edu
Jenny Valentine Nehring, *Utah Valley University*, jenny.nehring@uvu.edu

Abstract

This paper presents a validation-oriented framework for automated cybersecurity dataset labeling that separates initial event detection from final label assignment. Common labeling techniques in intrusion detection systems typically rely on single points of validation and are prone to noise and errors. To address this problem, the proposed framework collects initial and subsequent validation signals to determine whether candidate labels should be accepted, reversed, or escalated for review. The framework treats authentication outcomes and blocking events as candidate labels rather than ground truth and applies secondary validation techniques to reduce label noise before the labels are used in downstream analysis. Initial machine learning (ML) experiments demonstrate that the collected data is usable for downstream analysis and supports adaptive, environment-specific cybersecurity modeling.

Keywords: Cybersecurity, Intrusion Detection Systems, Honeypots, Continuous Learning, Automated Data Labeling, Network Traffic Analysis

Introduction

Autolabeling is common in modern honeypot architectures, but generating realistic synthetic benign behavior is difficult in honeypot and simulated environments. Production environments contain both malicious and benign activity from which data for artificial intelligence (AI)-powered intrusion detection system (IDS) models can be collected. Real-world human interaction creates noise in these processes due to human errors and successful attacks which can introduce logical flaws in modeling benign and malicious traffic based on a single-layer validation. This paper addresses problems with faults introduced in single-validation paths and introduces mechanisms to increase reliability in data autolabelling through low-impact, secondary means.

The Relevant Data Problem

Data relevance is also a significant challenge when using AI in cybersecurity. Public data models are typically used when addressing novel data questions (Kenyon et al., 2020). These public datasets are often antiquated or irrelevant to emerging cybersecurity problems (Brown et al., 2009; Kenyon et al., 2020; McHugh, 2000). For modern IDS to be applicable, they should process dynamic data during system operation to account for operational drift in cyber threats and in signals (Vassilev et al., 2024). Additionally, comprehensive datasets require powerful hardware to process their expansive nature and may carry bias (Zaid & Garai, 2024). Synthetic data is also problematic because it does not describe real-world threats, but it is often infeasible to differentiate in real-world scenarios using log data, even when log data is found to be superior as a classifier (Landauer et al., 2023).

Contributions

This paper contributes a validation-oriented framework for automated cybersecurity dataset labeling. The framework treats authentication outcomes, blocking events, and session metadata as candidate-label evidence rather than ground truth. It creates a useful framework for providing site-specific defensive structures without relying on third-party datasets by advocating self-hosted AI analysis. It then applies secondary validation, including multi-factor authentication (MFA) outcomes, silent-alarm behavior, and support escalation, to reduce false benign and false malicious labels before the data is used for model training. Lastly, we model the effects of errors in labeling data to demonstrate the impacts of false positive and false negative errors using a highly accurate training model with introduced defects. This approach supports adaptive, environment-specific cybersecurity datasets while preserving the distinction between initial event detection and final label assignment. The contribution of this work is a validation-oriented labeling framework, not a new IDS data model or attack-classification algorithm.

Related Work

A common approach to cybersecurity anomaly-detection research is to configure honeypots, manually generate attack data, manually label the resulting activity, and then evaluate detection models using the labeled data (Alatawi & Albalawi, 2025; Nimmagadda & Mehr, 2025). However, reliance on pre-existing labeled datasets can limit the usefulness of IDS models because public datasets may become outdated, may not reflect current threats, and may not represent the benign behavior of a specific production environment.

Several existing approaches attempt to address the labeling problem directly. Guerra et al. (2022) argue that labeling representative network traffic is difficult, expensive, and often dependent on specialized expertise. They note that many current methods rely on synthetic traffic generation or limited public datasets, which may become outdated and may fail to represent the distinction between normal and malicious behavior in current environments. Landauer et al. (2022) propose an automatic labeling framework for host-based IDS log datasets generated from model-driven testbeds. Their approach improves reproducibility by generating controlled log datasets with known attack steps and labeled events. However, both approaches leave open the problem of validating labels in production-like environments where authentication failures, legitimate user mistakes, blocking events, and compromised access may overlap. In contrast, the framework proposed in this paper treats authentication outcomes, blocking events, and packet metadata as candidate-label evidence rather than ground truth. Final labels are assigned only after secondary validation signals, such as MFA outcomes, silent-alarm behavior, session correlation, and support escalation, are evaluated.

Both rule-based intrusion prevention and ML-based intrusion detection are commonly used for evaluating secure shell (SSH) brute-force activity. Fail2Ban is widely used as a lightweight defensive log-based IDS tool that bans IP addresses after repeated failures using predefined filters and actions. However, because Fail2Ban is threshold-driven, adversaries may adapt their behavior around the rules, and legitimate users may be falsely identified after repeated authentication mistakes. Recent work comparing ML strategies with Fail2Ban and PAM Tally2 found that traditional tools can detect aggressive attacks but may miss slower or distributed attack patterns and may also produce false positives (George & Zeratsion, 2025). Other automatic labeling approaches have also been explored, but many rely on controlled testbeds, honeypot assumptions, or rule templates that do not fully address label accuracy in production-like environments (Guerra et al., 2022; Landauer et al., 2022; Satpute et al., 2025).

Our prior work in honeypot architecture also demonstrates the utility of leveraging production systems alongside honeypots to derive advanced persistent threats (Loper et al., 2025). You et al. (2022) use

emerging threat data to identify threat actors from unstructured sources, but such data may lack the contextual labeling needed for supervised dataset generation.

Recent work also highlights the importance of label noise and operational change. Zhao et al. (2026) introduce a graph-based selection and purification process for reducing label noise in network intrusion detection, while Martins (2020) addresses continuous data, sliding windows, and concept drift in cybersecurity data streams. These concerns are important for long-term cybersecurity modeling because both attack behavior and legitimate user behavior change over time.

Taken together, prior work addresses important parts of the problem, including honeypot-based data collection, testbed-generated datasets, log-based labeling, label-noise reduction, and ML-based intrusion detection. However, these approaches do not fully address the challenge of producing trustworthy labels from production-like systems where authentication failures, user mistakes, blocking events, and confirmed malicious behavior may overlap. This paper addresses that gap by treating authentication and blocking events as candidate labels that must pass secondary validation before being used as confirmed benign or malicious labels.

Methodology

Our method implements a validation pipeline that converts authentication and blocking events into finalized dataset labels using multi-stage verification. We also collected data for a known, high-confidence data feature in production data within our honeypot architecture to validate the data flow process and to introduce errors to measure performance. This work is constrained to an SSH-focused scope and a platform-biased approach, since validating attackers' threat environments is not feasible. Therefore, benign and malicious traffic were generated from different client environments and subsequently validated using a limited production SSH honeypot deployment that provided implicit label distinctions between legitimate and malicious authentication activity.

Validation Path

The validation logic is modeled as a Boolean decision process that promotes candidate labels into trusted dataset labels. Authentication outcomes, blocking events, MFA results, silent-alarm behavior, and support escalation outcomes are evaluated jointly to determine final label assignment. In all training models, a partition between training and test sets was performed, and a seed for this partition was used to create iterations of each test. The results given are an average of all tests.

Figure 1 illustrates the validation pipeline used to transform authentication and operational signals into final binary labels. The model is driven by a combination of signals, including authentication outcomes, repeated failed login thresholds, blocking events, MFA results, silent-alarm responses, and support escalation decisions. These signals are composed into candidate benign and malicious labels. Successful authentication contributes to a benign label path, and failed or anomalous behavior contributes to a malicious label path. However, no individual signal is treated as definitive; instead, the model applies strict logical rules that enforce the binary nature of the final label. A benign label is assigned only when all required validation conditions are satisfied or when support explicitly resolves the event as legitimate, while a malicious label is assigned when one or more failure conditions persist without reversal. This creates a deterministic decision boundary in which each event is ultimately classified as either benign or malicious, but only after passing through the validation pipeline. The pipeline ensures that raw operational events are first treated as provisional signals, then evaluated through layered validation logic, and finally promoted to

trusted dataset labels suitable for downstream analysis and model training. A process-flow view of this same data path is found in Figure 2.

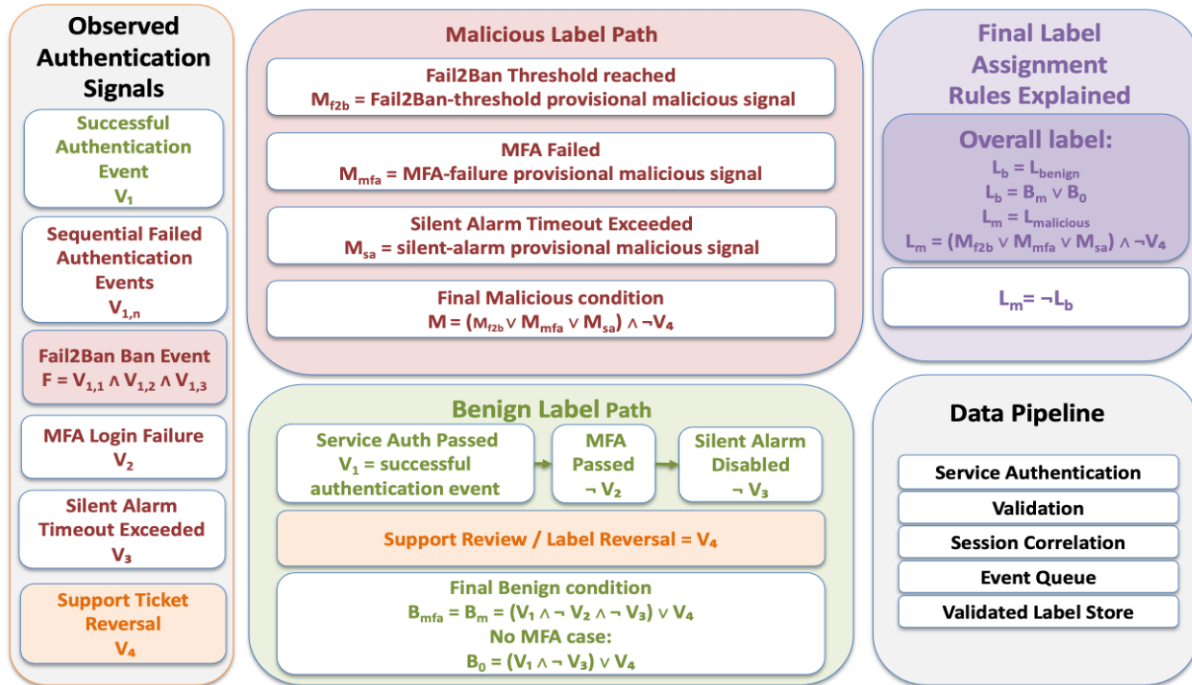


Figure 1. Validation Logic for Automated Label Assignment.

Let:

- L_m = final malicious label,
- L_b = final benign label,
- V_1 = successful authentication,
- $V_{1,n}$ = failed authentication attempt at sequence n,
- M_{mfa} = MFA failure,
- M_{sa} = silent alarm timeout,
- V_4 = support-based label reversal,
- F = failed-authentication threshold reached.

The benign condition is defined as:

$$L_b = \begin{cases} (V_1 \wedge \neg M_{mfa} \wedge \neg M_{sa}) \vee V_4, & \text{MFA case} \\ (V_1 \wedge \neg M_{sa}) \vee V_4, & \text{No MFA} \end{cases}$$

The malicious condition is defined as:

$$L_m = (F \vee M_{mfa} \vee M_{sa}) \wedge \neg V_4$$

This formulation ensures that no single signal determines a final label. Instead, labels are assigned only after evaluating all available validation outcomes.

Validation Event Collection

Table 1 outlines the primary labeling scenarios and highlights where false positives and false negatives may occur during authentication-based labeling. False positives can arise when legitimate users trigger repeated failed authentication thresholds or incomplete validation steps, while false negatives may occur when compromised access successfully passes initial authentication checks. To address these risks, all candidate labels are treated as provisional and are subject to secondary validation, including MFA outcomes, silent-alarm responses, and support escalation. This process enables the system to correct mislabeled events by reversing or confirming labels before they are finalized for dataset inclusion.

Table 1: *Authentication Outcomes and Candidate Label Validation Rules*

Authentication Outcome	Candidate Label	Validation Concern	Validation Response
Successful authentication	Candidate benign label	Possible compromised access	Require MFA, silent-alarm clearance, or support resolution
Successful authentication with failed validation	Candidate malicious label	Incomplete validation	Retain suspicious state or escalate
Repeated failed authentication threshold reached	Candidate malicious label	User error possible	Treat as provisional pending review
Repeated failed authentication resolved by support	Final benign label	Confirmed user error	Reverse provisional label

Not all possible interactions with the system are collected. Events are collected only when they are relevant to labeling. Specifically, when they are successful during authentication or logged as repeated failed authentications. Excluded are edge cases, such as a connection from a host where authentication is not attempted or when the limit to authentication attempts is not exceeded. These events should be documented and individually analyzed before a judgment on a label can be decided. Each logged event is paired with packet-capture metadata to enable session-level correlation during validation and review.

Secondary validation signals, including MFA results and silent-alarm outcomes, are incorporated as independent evidence. These signals provide additional context for confirming or rejecting candidate labels.

Data Structures

Validated labels are stored with associated session metadata, including authentication outcomes, blocking events, timestamps, and packet-capture features. This enables correlation across validation signals and supports downstream ML analysis.

Validation-Oriented Auto-Labeling Framework for Cybersecurity Datasets

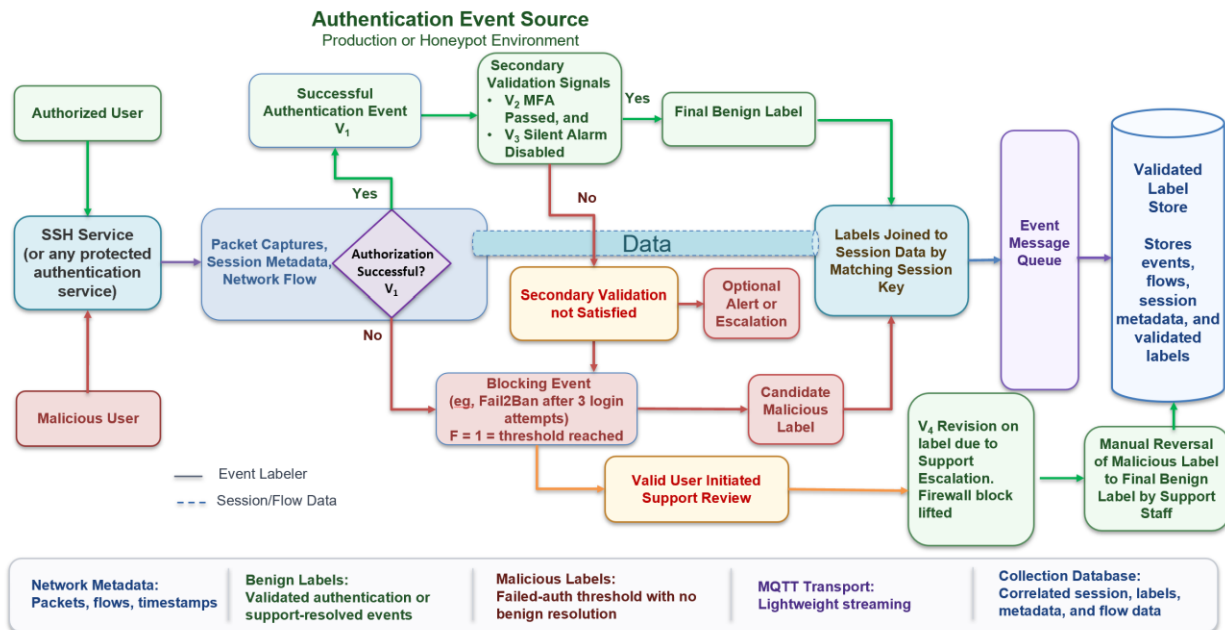


Figure 2. Validation-Oriented Auto-Labeling Framework for Cybersecurity Datasets.

Research Questions

1. Can combined authentication and secondary validation signals improve the confidence of authentication-based candidate labels?
2. Does the introduction of labeling defects produce a measurable and generally monotonic effect on accuracy, precision, and recall?

Data Collection

For each session, the system logs authentication outcomes, blocking status, timestamps, network identifiers, packet metadata, and validation signals. These elements are used to evaluate and finalize candidate labels prior to inclusion in the dataset.

Experimental Setup and Implementation

A controlled cyber range was used to generate labeled SSH traffic. OpenSSH on Linux represented malicious activity, while PuTTY on Windows represented benign activity. A limited set of features was used to demonstrate the label validation process (see Table 2). The default TCP header length difference between platforms provided a strong discriminative feature. For example, packets generated by the Windows environment used a greater default TCP header length than packets generated by the Linux environment. In the generated dataset, this feature ranged from 32–40 bytes in the PuTTY-generated packets labeled as benign and from 20–32 bytes in the Linux OpenSSH packets labeled as malicious.

A dataset of 2,495 accurately labeled samples was collected. Models including Random Forest, Decision Tree, Logistic Regression, Gaussian Naïve Bayes (Naïve Bayes), and Gradient Boost were evaluated using 50 randomized seed runs to establish a baseline. We use a limited set of features available from tcpdump data. We then ran five randomized seed runs for each error level (0-20 percent). In other words, fifty runs

were used to establish baseline stability; five runs per corruption level were used because of the larger number of corruption conditions. The introduced errors represented labeling failures that could occur when the secondary validation steps proposed by the framework are absent. Label corruption was introduced incrementally at 1 percent per epoch to measure performance degradation. Naive Bayes performed poorly relative to other models in all cases.

Table 2: *Evidence Classification and Role in the Data Model.*

Evidence category	Examples	Role	Limitation
Authentication	Success, repeated failure	Candidate label	Compromised access or user error
Blocking	Fail2Ban event	Suspicion signal	Threshold-based false positives
Secondary validation	MFA, silent alarm	Confirm/reject candidate	Missing or evaded signal
Escalation	Verified support review	Resolve uncertainty	Cost and social engineering
Packet/session metadata	TCP header, flags, timestamps	Correlation and downstream modeling	Platform and stack bias

To further evaluate the proposed framework, a similar environment was deployed as a publicly accessible SSH honeypot exposed to live Internet threats. A honeypot was chosen rather than a production-only system because it allowed controlled generation of benign authentication events while simultaneously capturing authentic attack traffic. This provided a more reliable environment for evaluating the label-validation process than a production system alone. During a 16-hour collection period, the environment recorded authentication activity from both threat actors and legitimate users.

Results

The experimental results show that the collected authentication-based dataset was usable for downstream ML analysis using a comparison between the 50-seed run and the 5-seed run at the zero percent error level. Random Forest, Decision Tree, Logistic Regression, and Gradient Boost performed near 100 percent on the accurately labeled test data. This suggests that the dataset contained learnable signals, but not all model types were equally appropriate for the feature space (i.e., Naive Bayes).

The label-corruption experiments showed that model performance declined as mislabeled data increased. Figure 3 illustrates the behavior of the Gradient Boost model under both individual and combined benign and malicious label corruption, with shaded regions representing variation across five randomized seed runs. In general, all models exhibited approximately linear degradation in performance as label noise increased. One notable deviation occurred in precision for the Gradient Boost model under benign-label corruption, where the decline is less consistent. Additionally, recall exhibited a confounding effect when

both benign and malicious labels were corrupted simultaneously, deviating from the trends observed in the single-label corruption scenarios. This interaction suggested that simultaneous corruption across both labels introduces increasing variability in model behavior. This reinforces the need for validation mechanisms that address both benign and malicious labeling errors to maintain consistency in predictions.

Figure 4 extends this comparison across the tested models and corruption scenarios. The figure combines accuracy, precision, and recall to provide a unified comparison across models and corruption scenarios. The various panels illustrate the performance of the Decision Tree, Gradient Boost, Logistic Regression, Naive Bayes, and Random Forest algorithms, using blue for accuracy, orange for precision, and green for recall. The fuzziness in each of these colors reflects the instability across runs with randomized trials. As seen in these figures, Random Forest and Decision Tree exhibited comparatively stable performance and low variance across the five randomized seed runs.

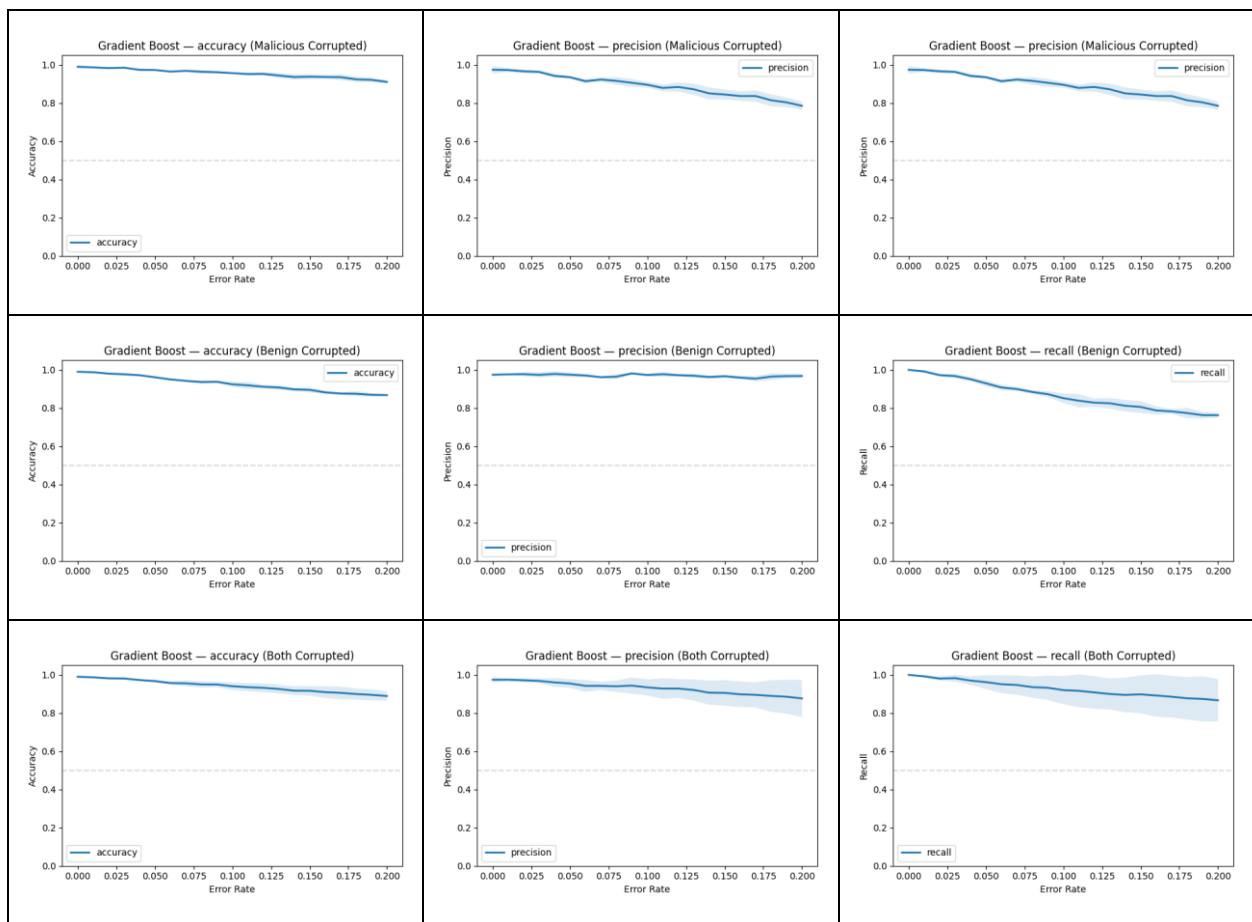


Figure 3. Details of Gradient Boost Accuracy, Precision, and Recall.

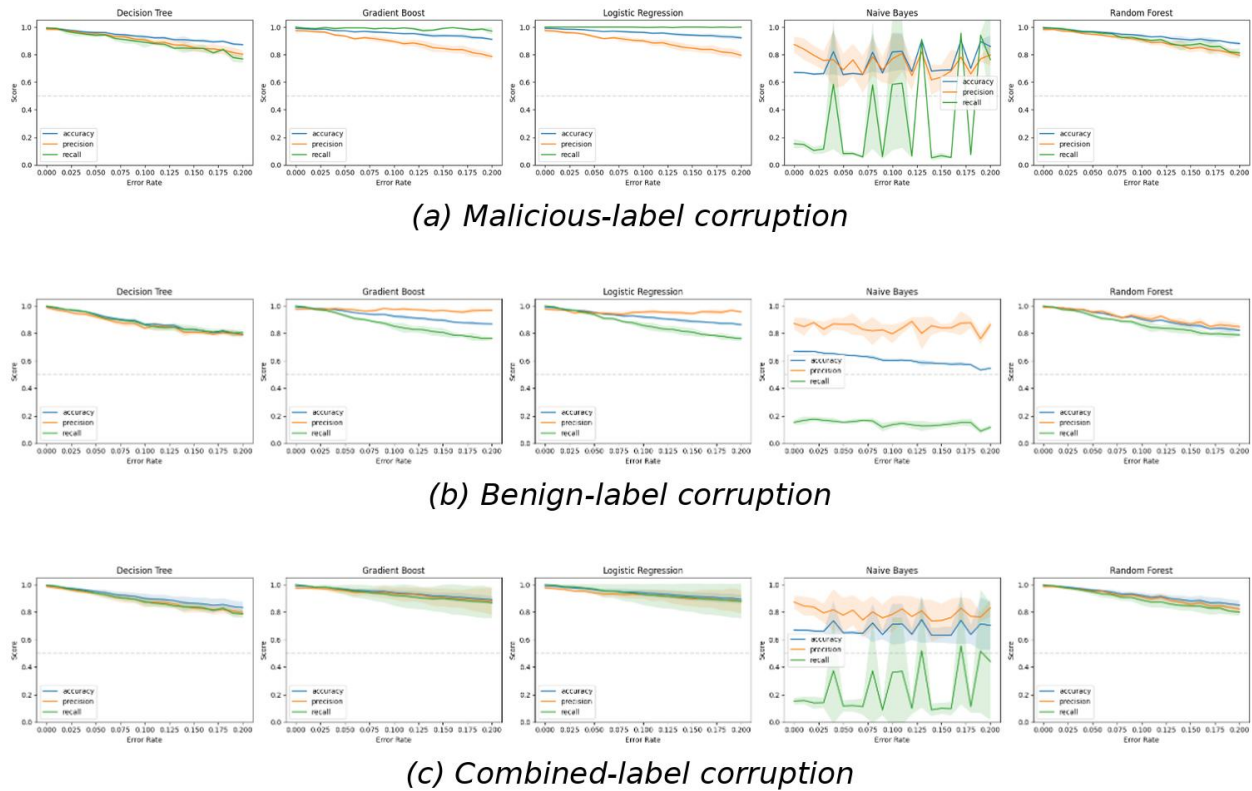


Figure 4. Effects of Label Corruption on Malicious, Benign, and Combined-Label Corruption Scenarios.

Table 3 summarizes the accuracy degradation at selected corruption rates. Random Forest performance declined to 0.947 (94.7 percent) when labels were corrupted to an 8 percent level, 0.915 when benign labels were corrupted, and 0.931 when both label classes were corrupted. Gradient Boost retained 0.964, 0.936, and 0.950 accuracy across the same three scenarios. Logistic Regression also remained relatively stable, while Decision Tree showed a larger decline when benign labels were corrupted. Naive Bayes remained substantially lower than the other models across most conditions.

Table 3: Model Accuracy Degradation Under Label Corruption Scenarios (Lab-only environment)

Scenario	0%	1%	2%	3%	5%	8%
Malicious Labels Corrupted						
Random Forest	0.992	0.991	0.985	0.978	0.968	0.947
Decision Tree	0.992	0.990	0.979	0.972	0.961	0.943
Logistic Regression	0.990	0.989	0.985	0.984	0.974	0.967
Naive Bayes	0.671	0.669	0.659	0.662	0.656	0.817
Gradient Boost	0.990	0.987	0.983	0.985	0.973	0.964
Benign Labels Corrupted						
Random Forest	0.995	0.991	0.983	0.976	0.953	0.915
Decision Tree	0.995	0.984	0.973	0.964	0.940	0.897
Logistic Regression	0.991	0.985	0.976	0.966	0.953	0.925
Naive Bayes	0.671	0.668	0.669	0.656	0.644	0.625
Gradient Boost	0.990	0.988	0.980	0.977	0.961	0.936
Both Labels Corrupted						
Random Forest	0.994	0.991	0.984	0.977	0.960	0.931
Decision Tree	0.993	0.987	0.976	0.968	0.950	0.920
Logistic Regression	0.991	0.987	0.980	0.975	0.963	0.946
Naive Bayes	0.671	0.669	0.664	0.659	0.650	0.721
Gradient Boost	0.990	0.987	0.982	0.981	0.967	0.950

Note. Values represent mean classification accuracy across five independent random-seed runs at each simulated label-corruption level.

The results also address the two research questions. For Research Question 1, the controlled operational scenarios demonstrate that combined authentication and secondary validation signals can improve label confidence by correcting specific false-benign and false-malicious candidate labels. However, comparison against independently reviewed ground truth is required to quantify the framework's overall labeling accuracy.

For Research Question 2, the results indicate that the introduction of labeling defects produces a predictable degradation in model performance. This includes declines in accuracy, precision, and recall. As the rate of label corruption increases, all evaluated models exhibit consistent performance loss, with the magnitude and rate of degradation varying by model and corruption type. In particular, corruption of benign labels results in more pronounced declines in several models, demonstrating that errors in labeling systematically weaken the relationship between features and class assignments. These findings confirm that labeling defects have a measurable and predictable negative impact on model performance metrics.

These results support the central argument of the framework: authentication outcomes and Fail2Ban events should be treated as candidate labels rather than ground truth, since it is possible for a valid user to be blocked by Fail2Ban and for a malicious user to be authenticated in a successful attack. Even modest label corruption reduced model performance, which demonstrates the importance of secondary validation before labels are stored for training or evaluation. These scenarios illustrate failure modes of authentication-only and Fail2Ban-only labeling; they are not intended as a statistical comparison against those baselines.

To evaluate the framework under operational conditions, we applied the laboratory-trained model to a production SSH honeypot environment. Anonymous attackers were allowed to use their own tools and techniques against the production system, while benign authentication events were generated through normal SSH logins using PuTTY on Windows, consistent with the laboratory methodology. Although the

production environment introduced concept drift due to uncontrolled attacker behavior, the model continued to perform well, achieving 96.52% accuracy and recall with 99.96% precision. These results indicate that models trained in controlled environments remain effective when deployed to production, although some performance degradation should be expected as attacker behavior diverges from laboratory conditions.

We next trained a Random Forest model using data collected from the production SSH honeypot environment. During a 16-hour period, the honeypot recorded attacks from 683 unique IP addresses. Production benign sessions were generated by authorized researchers, while malicious sessions were limited to unsolicited Internet-originating activity against the dedicated honeypot; packet labels inherited the validated session label. The resulting model achieved performance comparable to that of the laboratory-trained model. The production dataset contained 210,031 malicious packets and 9,561 benign packets. To reduce the effects of class imbalance during analysis, the malicious class was randomly downsampled to produce an approximately 2:1 malicious-to-benign ratio. Table 4 shows the lab model performance in the lab data, the lab model on production data, and the production model on production data, demonstrating deployment-related distribution shifts.

Table 4. *Comparison of Laboratory- and Production-Trained Model Performance.*

Metric	Lab Model on Lab Data	Lab Model on Production Data	Production Model on Production Data
Accuracy	99.2%	96.5%	99.0%
Precision	98.3%	99.9%	99.9%
Recall	99.8%	96.5%	99.1%

Finally, we evaluated the framework using controlled false-positive and false-negative scenarios. To produce a false-negative candidate label, an attacker successfully authenticated using a deliberately weak account (username: admin; password: password). Although the successful authentication initially produced a candidate benign label, the attacker's failure to clear the silent alarm triggered secondary validation and reclassified the session as malicious (V_2 or V_3 in Figures 1 and 2). This scenario demonstrated how the proposed validation process correctly reclassified the event from benign to malicious. To simulate a false positive, we intentionally generated failed SSH authentication attempts using PuTTY with password authentication. These events were initially labeled as malicious but were subsequently reclassified through the escalation workflow (V_4 in Figures 1 and 2) after manual verification and removal of the Fail2Ban block.

Table 5. *Production Model Accuracy Under Introduced Label Corruption.*

Scenario	0%	1%	2%	3%	5%	8%
Malicious Label Corruption	0.9997	0.9965	0.9931	0.9899	0.9832	0.9731
Benign Label Corruption	0.9997	0.9931	0.9866	0.9798	0.9666	0.9466
50/50 Mal/Benign Label Corruption	0.9997	0.9950	0.9899	0.9851	0.9750	0.9600

Discussion

The results show that label quality strongly affects the performance of cybersecurity models trained on production data (see Table 5). Models perform well with accurate labels but degrade as label noise increases, indicating that improving validation pipelines can be as important as selecting more advanced models. Our findings demonstrate that simultaneous corruption of both label classes produced substantial variability in some ML models. Label accuracy refers to agreement with verified event identity, whereas model accuracy refers to downstream classification performance on the resulting dataset.

The findings also highlight the need to separate detection from final labeling. Tools like Fail2Ban provide useful blocking signals, but blocks do not necessarily indicate malicious activity. Legitimate users can trigger lockouts through errors or expired credentials. Without secondary validation, these events risk being mislabeled. The proposed framework mitigates this through MFA outcomes, silent alarms, session correlation, and support escalation to confirm, reverse, or finalize labels.

Both the laboratory and production honeypot environments demonstrate the importance of generating cybersecurity datasets from contemporary authentication and blocking events to account for deployment-related distribution shifts and concept drift. The laboratory environment assumed that benign users authenticated from Windows systems, while attackers primarily used Linux-based tools. Although this provided an effective baseline model, performance declined slightly when the model was deployed in production, where attacker behavior differed from those controlled assumptions. Despite this degradation, the model continued to perform well, suggesting that the laboratory assumptions (including the likelihood of Linux use) captured many characteristics of real-world attacks, even though production attacker behavior was more diverse and led to a 4% reduction in performance.

Training a new model using production data restored much of the lab's initial performance, demonstrating the value of continuously adapting models to evolving operational environments to account for concept drift. Furthermore, the production validation framework illustrates how well-curated datasets can be generated from operational data and how label quality degrades when validation mechanisms are absent.

Benign-label corruption proved particularly detrimental. Several models experienced greater performance degradation from benign-label corruption than from malicious-label corruption, indicating that normal user activity—including retries, credential expiration, and automated defensive responses—is especially susceptible to misclassification. A validation framework that corrects provisional labels can reduce false positives and improve the quality of cybersecurity datasets.

The framework is practical for production use, leveraging existing logs and metadata without requiring full manual labeling. It generates candidate labels from operational signals and validates them before training, enabling environment-specific datasets while preserving the distinction between detection and verified labels.

Limitations

This study was conducted in both a controlled cyber range and a production SSH honeypot environment exposed to real-world threats. The distinction between Windows PuTTY traffic and Linux OpenSSH traffic produced a strong classification dataset because of the highly correlated default TCP header length differences between the two environments. The production results indicate that the platform-correlated distinctions learned in the laboratory remained prevalent in the observed deployment. They do not establish

that the classifier learned attack behavior independently of client or network-stack characteristics. Because TCP header characteristics can be modified with relatively little effort, sophisticated adversaries could potentially evade classifiers that rely heavily on these enumerated features. Production deployments should incorporate broader and evolving feature sets, continuous monitoring for concept drift, and a defined model-retraining schedule. Therefore, future research work should evaluate this proposed framework using more diverse traffic sources, additional network services, deeper packet-level features, and longer production deployments. Furthermore, the experiments in this study focused exclusively on SSH authentication traffic. Although the proposed label-validation framework is designed to be applicable beyond SSH, additional evaluation using other authentication services (e.g., RDP, web applications, VPNs, and cloud identity providers) is needed to determine its broader applicability. Accordingly, we emphasize that the primary contribution is the label-validation architecture rather than the specific classifier.

The current implementation also limits candidate labels to events generated by successful authentication or Fail2Ban blocking actions. Consequently, only IP addresses that reached the blocking threshold were initially labeled as malicious, while benign sessions were identified through successful authentication occurring within three or fewer attempts. An attacker could instead perform only one or two failed authentication attempts before pivoting to a different IP address, thereby avoiding the validation signals used by the framework. However, because attackers generally cannot determine whether the proposed validation framework has been deployed, aggressively rotating through large numbers of proxy IP addresses after only one or two attempts provides uncertain benefit while substantially increasing operational complexity. Future work should investigate additional candidate validation signals capable of identifying malicious activity before authentication success or blocking events occur.

Security Considerations

The framework remains susceptible to attacker-driven data poisoning, evasion of labeling mechanisms, and honeypot fingerprinting. Although these risks also affect conventional authentication- and threshold-based labeling systems, the proposed framework introduces additional validation signals that adversaries may attempt to manipulate or evade. Conflicting signals, missing validation data, and uncertain outcomes may also prevent automatic label finalization. In such cases, the event should remain unresolved or be escalated for manual review rather than being forced into a benign or malicious class.

The V_4 support-review mechanism shown in Figures 1 and 2 is particularly influential because it can reverse a provisional label. However, the human component is potentially vulnerable to social engineering, inconsistent judgment, and documentation errors. Organizations should therefore require analyst training, authenticated support requests, auditable reversal decisions, and role-based authorization for manual label changes.

Implications and Conclusions

We showed how production environments can be used to collect high-quality, relevant datasets for site-specific ML modeling. Because these data are narrowly tailored to attacks against site-specific services, the models for defense are timely and lightweight. Thus, they are relevant and useful for many network edge and IoT considerations.

The results show that machine learning models performed well when labels were accurate, but performance declined as label corruption increased. This confirms that label quality directly affects model reliability. By separating event detection from final label assignment, the proposed framework provides a practical method for improving the trustworthiness of cybersecurity datasets generated from authentication activity, blocking events, and packet metadata. To support reproducibility and future research, the implementation and supporting materials used in this study are available in our public GitLab repository: <https://gitlab.com/indigeos/uvu-ist-research>.

Future Work

Future implementations should include stronger disagreement handling, monitoring for label drift, and safeguards against adversarial manipulation of the labeling process. Future work in this domain should evaluate the framework in larger and more diverse environments. This includes longer production deployments, additional authentication services, broader packet and flow-level features, and comparison against manually reviewed ground truth. Additional work should also examine concept drift, attacker poisoning, label-disagreement handling, and the integration of MFA and support-ticket data into the validation pipeline.

Additional future work would analyze extensive, real adversary traffic collected from Internet-exposed production environments, including Internet-exposed honeypot traffic, production authentication attempts, real attacker behavior, and longitudinal data collection. This expansion should also allow robust testing for authentication-only labels, Fail2Ban-only labels, manual analyst labels, SIEM-derived labels, and other automated labeling frameworks.

These aims should provide a foundation for a general improvement in robustness of the validation system that will allow us to study how labels and features change over time. These would include analysis of concept drift, label drift, attacker adaptation, and optimal model retraining intervals. This would also allow the evaluation of attacks against the validation framework, including attempts at data poisoning, label poisoning, honeypot fingerprinting, low-and-slow attacks, and validation-signal evasion.

By adding production elements and expanding deployment into production environments, additional client behaviors and heuristics can be explored. It is necessary, when practicable, to log client operating system information and SSH client application and version numbers. Real-world benign traffic will primarily be collected from production systems, whereas honeypot deployments will continue to provide high-confidence malicious traffic.

In addition to a greater variety of client environments to analyze, future work should expand beyond SSH to include other services like web authentication, VPN authentication, email authentication, SFTP (secure file transfer protocol), remote administration services, and enterprise SSO. Operational robustness should also be expanded to include analysis of storage overhead, workload, escalation volume, maintenance burden, scalability, data-retention policy, data privacy, and compliance issues.

References

- Alatawi, E., & Albalawi, U. (2025). Harnessing AI for cyber defense: Honeypot-driven intrusion detection systems. *Symmetry*, 17, 628.
- Brown, C., Cowperthwaite, A., Hijazi, A., & Somayaji, A. (2009). Analysis of the 1999 DARPA/Lincoln Laboratory IDS evaluation data with NetADHICT. In 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications (pp. 1-7).
- George, R., & Zeratsion, E. (2025). *Detection of SSH brute-force attacks using machine learning: A comparative study with Fail2Ban and PAM Tally2* [Master's thesis, University West].
- Guerra, J. L., Catania, C., & Veas, E. (2022). Datasets are not enough: Challenges in labeling network traffic. *Computers & Security*, 120, 102810. <https://doi.org/10.1016/j.cose.2022.102810>
- Kenyon, A., Deka, L., & Elizondo, D. (2020). Are public intrusion datasets fit for purpose characterising the state of the art in intrusion event datasets. *Computers & Security*, 99, 102022.
- Landauer, M., Frank, M., Skopik, F., Hotwagner, W., Wurzenberger, M., & Rauber, A. (2022). A framework for automatic labeling of log datasets from model-driven testbeds for HIDS evaluation. In Proceedings of the 2022 ACM Workshop on Secure and Trustworthy Cyber-Physical Systems (pp. 77-86).
- Landauer, M., Skopik, F., Frank, M., Hotwagner, W., Wurzenberger, M., & Rauber, A. (2023). Maintainable log datasets for evaluation of intrusion detection systems. *IEEE Transactions on Dependable and Secure Computing*, 20(4), 3466-3482.
- Loper, D. C., Nehring, J. V., & Ranganathan, P. (2025). Vast, low-cost honeypot architecture. *Journal of Computing Sciences in Colleges*, 41(2), 195-210.
- Martins, M. I. P. B. (2020). *Anomaly detection in cybersecurity* [Master's thesis, Universidade do Porto].
- McHugh, J. (2000). Testing intrusion detection systems: A critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory. *ACM Transactions on Information and System Security*, 3(4), 262-294.
- Nimmagadda, A., & Mehr, S. Y. (2025). AI-powered intrusion detection system with honeypot integration. *International Journal of Intelligent Information Systems*, 14(4), 70-76.
- Satpute, A., Nikam, S., Gaikwad, V., Kakade, Y., & Mhaske, C. (2025). AI-driven intrusion detection system using SSH honeypots. *ICCK Transactions on Cybersecurity*, 1(1), 3-12.
- Vassilev, A., Oprea, A., Fordyce, A., & Anderson, H. (2024). Adversarial machine learning: A taxonomy and terminology of attacks and mitigations (Technical Report NIST 100-2e2023). National Institute of Standards and Technology.
- You, Y., Jiang, J., Jiang, Z., Yang, P., Liu, B., Feng, H., Wang, X., & Li, N. (2022). TIM: Threat context-enhanced TTP intelligence mining on unstructured threat data. *Cybersecurity*, 5(1), 3.

Zaid, T., & Garai, S. (2024). Emerging trends in cybersecurity: A holistic view on current threats, assessing solutions, and pioneering new frontiers. *Blockchain in Healthcare Today*, 7, 10.30953/bhty.v7.302.

Zhao, R., Ding, J., Dong, Q., & Cheng, H. (2026). Mitigating label noise in network intrusion detection via graph-based sample selection and purification. *Scientific Reports*, 16(1), 11674.