Cross Dataset Evaluation of Robustness in Machine Learning Based Intrusion Detection Under Structured Label Poisoning

Damla Görgülü Eren Yavuz Hakan Çapuk Oğuz Kağan Hitit Rana Ataseven

1 Summary

Machine-learning–based Network Intrusion Detection Systems (NIDS) are widely deployed to detect malicious activity in high-volume networks, yet their robustness to training-time label corruption remains underexplored. In practice, adversaries may exploit data collection and labeling pipelines to introduce targeted label noise that selectively suppresses detection of high-value intrusions or distorts decision boundaries in critical regions of feature space. Despite the operational relevance of such structured label-poisoning threats, existing evaluations are often limited to single datasets or narrow model classes, leaving cross-dataset vulnerability poorly characterized.

This project will address this gap through a systematic, cross-dataset study of targeted label-poisoning attacks and practical defenses for NIDS. We will evaluate five complementary benchmarks, UNSW-NB15, LYCOS-IDS2017, CUPID, and CIDDS-001, using a representative model suite spanning linear and ensemble baselines and neural architectures (Logistic Regression, Random Forests, MLP, 1D-CNN and RNN). Under bounded label-flip budgets, we will implement structured poisoning strategies including class-hiding, feature-targeted, confidence-/loss-aware, disagreement-based and temporal-window and quantify robustness using accuracy, macro/per-class precision–recall–F1, confusion matrices, and degradation curves versus poisoning rate. We will further compare training-time defenses based on loss- and disagreement-driven filtering and reweighting, and systematically characterize how these defenses affect poisoned-data robustness and clean-data performance across datasets, model families, and attack strategies.

2 Technical Approach

Our technical approach comprises four components: dataset preparation, model development, attack design, and defense with evaluation. All experiments will be implemented in Python using scikit-learn and PyTorch, with a modular pipeline so that datasets, poisoning strategies, models, and defenses can be combined and reproduced consistently.

2.1 Datasets and Preprocessing

Prior work has shown that performance conclusions derived from a single NIDS dataset often fail to generalize across differing traffic collection methodologies, attack taxonomies, and feature extraction pipelines, thereby motivating cross-dataset evaluation as a more reliable assessment practice [?]. Recent survey work further systematizes existing intrusion detection datasets and identifies structural limitations that impede comparative evaluation and generalization across studies [?]. Guided in part by this framework and its dataset-selection recommendations, we evaluate our approach on five recent, complementary NIDS benchmarks spanning diverse collection methodologies, labeling reliability, and attack realism as summarized in Table 1.

Table 1: Summary of evaluated NIDS datasets and selection rationale.

Dataset	Traffic Characteristics	Attack Coverage	Rationale for Selection
UNSW-NB15 [?]	Controlled cyber-range traffic (PCAP + bidirectional flows)	9 attack categories	Widely used baseline benchmark
LYCOS-IDS2017 [?]	CIC-IDS2017 re-extracted and corrected flows (LycoSTand)	Brute-force, DoS/DDoS, botnet, web, infiltration, heartbleed	Mitigates CIC-IDS2017 flow and feature artifacts
CUPID [?]	Small testbed; professional pentesting (scripted + human); PCAP + features	Benign vs. attack (binary labels)	Captures realistic red-team behavior
CIDDS-001 [?]	Mixed real and emulated enterprise traffic; NetFlow features	Brute-force, DoS/DDoS, port scanning, infiltration	Provides enterprise-style flow data with explicit labels for robustness analysis

We will apply a unified preprocessing pipeline across datasets that removes malformed records and duplicate flows, encodes categorical fields (e.g., protocol, service, and flag) using one-hot encoding or learned embeddings as appropriate, normalizes numerical features, and constructs stratified training, validation, and test splits. Our proposed methodology for poisoning is discussed in Section **??**.

2.2 Model Families and Baselines

We evaluate a focused set of tabular classification models representing distinct inductive biases to study robustness under structured label-poisoning attacks. Our baselines include Logistic Regression (LR) as a linear, interpretable reference and Random Forests (RF) as a strong nonlinear ensemble method commonly used in intrusion detection. We further include a Multi-Layer Perceptron (MLP) as a standard neural baseline for tabular data, a 1D-CNN architecture and a RNN-style model that processes the data in a sequential manner to capture over-time dependencies. All hyperparameters are tuned on clean validation sets to ensure fair comparison.

2.3 Targeted Poisoning Strategies

We will evaluate structured training-time label poisoning under a bounded label-flip budget. The adversary will be limited to flipping a fraction of training labels, with no control over the model or optimizer, consistent with prior work [**?**, **?**].

We will study training-time label poisoning under a bounded threat model in which an adversary flips at most $\rho\%$ of training labels (features unchanged) while validation and test sets remain clean. For each dataset and $\rho \in \{0, 5, 10, 20\}$, we will generate poisoned training variants and retrain each model to obtain performance–poisoning curves. Within the fixed flip budget, we will evaluate the targeted strategies in Table **??**.

Table 2: Targeted label-poisoning strategies evaluated under a flip-budget ρ.

Strategy	Flip selection rule (budget ρ)	Intended effect
Class-hiding poisoning [?]	Flip labels from a target attack class ($\rightarrow$) benign (or majority) class	Suppress detection of a chosen attack class
Feature-targeted poisoning	Flip labels only for samples matching feature predicates (e.g., protocol/port/duration ranges)	Concentrate corruption in specific regions of feature space
Influence-/loss-aware poisoning [?]	Train a baseline model, then flip labels for highest-loss or lowest-confidence points (optionally guided by influence estimates)	Distort decision boundaries efficiently under a fixed budget
Disagreement-based poisoning [?]	Train two heterogeneous models, flip labels where predictions disagree most	Target ambiguous points likely to affect multiple model families
Temporal-window poisoning [?]	Sort flows by time, flip labels within selected contiguous windows	Time-localized corruption resembling campaigns or bursts

2.4 Defenses and Evaluation Protocol

To assess robustness to structured, training-time label poisoning, we will compare several training-time defenses under identical label-flip budgets across datasets, model families, and attack strategies. We will report (i) performance under poisoning and (ii) clean-data performance to quantify robustness–accuracy trade-offs and any adverse side effects on baseline detection quality.

2.4.1 Training-time defenses

We will implement and compare:

- **Poison-filtering (loss/confidence/disagreement signals).** We will flag suspicious training points using model-derived signals (e.g., high loss, low confidence, or persistent ensemble disagreement) and then either remove them or relabel them before retraining.

- **Reweighting instead of removal.** As a softer alternative, we will down-weight suspicious samples rather than discarding them, using a validation-driven or loss-driven weighting scheme.
- **Composite suspiciousness.** We will also evaluate a single combined ranking score that merges loss, confidence, and disagreement, then apply either filtering or reweighting to the top-ranked fraction. This is introduced as an implementation choice to unify the above signals (not claimed as a specific prior method).

2.4.2 Evaluation metrics and protocol

For each dataset, model family, poisoning strategy, and defense, we will generate poisoned training sets at multiple poisoning rates and retrain models from scratch, while keeping validation and test sets clean, matching the standard poisoning-game setup where an attacker contributes an ϵn-sized poisoned component and the defender trains normally on clean+poisoned data [**?**]. We will report overall accuracy and macro-F1, plus per-class precision/recall/F1 and confusion matrices to diagnose which attack categories degrade most. For each configuration, we will construct performance-versus-poisoning-rate curves consistent with label-flipping evaluations across poisoning rates.

3 Deliverables

- **Project report:** A final report describing the threat model, poisoning strategies, defenses, experimental protocol, and cross-dataset results with degradation curves and key ablations.
- **Reproducible source code repository:** Python implementation covering preprocessing, model training, poisoning generators, defense modules, and evaluation scripts.

4 Timeline

Table 3: Tentative project timeline and task ownership.

Task	**Dec 22–28**	**Dec 29–Jan 4**	**Jan 5–11**	**Jan 12–18**	**Jan 19–26**
Datasets, preprocessing, splits	Damla, Eren, Oğuz, Rana	Damla, Rana			
ML Model setup	Hakan	Damla, Eren, Hakan	Eren, Hakan		
Implementation of Poisoning Strategies		Hakan, Oğuz	Damla, Eren, Hakan		
Implementation of Defenses		Oğuz, Rana	Oğuz, Rana	Oğuz	
Experiments and Ablations				Damla, Eren, Hakan, Oğuz, Rana	Eren, Oğuz
Report writing				Damla, Hakan, Rana	Damla, Eren, Hakan, Oğuz, Rana

5 Bibliography

[1] R. Sommer and V. Paxson, "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection," in *IEEE Symposium on Security and Privacy*, 2010, pp. 305–316, doi: 10.1109/SP.2010.25.

[2] P. Goldschmidt and D. Chudá, "Network Intrusion Datasets: A Survey, Limitations, and Recommendations," *arXiv preprint* arXiv:2502.06688, 2025.

[3] N. Moustafa and J. Slay, "UNSW-NB15: A Comprehensive Data Set for Network Intrusion Detection Systems," in *Military Communications and Information Systems Conference (MilCIS)*, 2015, pp. 1–6, doi: 10.1109/MilCIS.2015.7348942.

[4] A. Rosay, F. Carlier, E. Cheval, and P. Leroux, "From CIC-IDS2017 to LYCOS-IDS2017: A Corrected Dataset for Better Performance," in *ACM International Conference Proceedings Series*, 2021, pp. 570–575, doi: 10.1145/3486622.3493973.

[5] J. Lawrence, G. Fahimipirehgalin, A. G. West, and A. Bardas, "CUPID: A Labeled Dataset with Pentesting for Evaluation of Network Intrusion Detection," *Journal of Systems Architecture*, 2022, doi: 10.1016/j.sysarc.2022.102621.

[6] M. Ring, S. Wunderlich, D. Grüdl, D. Landes, and A. Hotho, "Flow-based Benchmark Data Sets for Intrusion Detection," in *Proc. of the European Conference on Cyber Warfare and Security (ECCWS)*, 2017.

[7] Z. Wang *et al.*, "Threats to Training: A Survey of Poisoning Attacks and Defenses on Machine Learning Systems," *ACM Computing Surveys*, vol. 55, no. 7, 2023, doi: 10.1145/3538707.

[8] X. Chang, G. Dobbie, and J. Wicker, "Fast Adversarial Label-Flipping Attack on Tabular Data," *arXiv preprint* arXiv:2310.10744, 2023.

[9] P. W. Koh and P. Liang, "Understanding Black-box Predictions via Influence Functions," in *Proc. ICML*, 2017; arXiv:1703.04730.

[10] H. S. Seung, M. Opper, and H. Sompolinsky, "Query by Committee," in *Proc. COLT*, 1992, doi: 10.1145/130385.130417.

[11] L. Yang *et al.*, "CADE: Detecting and Explaining Concept Drift Samples for Security Applications," in *USENIX Security Symposium*, 2021.

[12] J. Steinhardt, P. W. W. Koh, and P. Liang, "Certified Defenses for Data Poisoning Attacks," in *Proc. NeurIPS*, 2017.