Available online at www.qu.edu.iq/journalcm
JOURNAL OF AL-QADISIYAH FOR COMPUTER SCIENCE AND MATHEMATICS
ISSN:2521-3504(online) ISSN:2074-0204(print)

Adversarial Robustness of Network IDS (Structured Data)

Haedar Ahmed Mukhef [a] , Hayder Hasan Ali [b] , Zahraa Sameer Ibrahim [a,b,*]

Mustansiriyah University Email: Info@uomustansiriyah.edu.iq

ARTICLE INFO

Article history:
Received: 1 /9/2025
Rrevised form: 19/10/2025
Accepted : 21 /10/2025
Available online: 30 /12/2025

Keywords:

Network Intrusion Detection;

Adversarial Robustness;

Cross-Domain Generalization;

Realizability Constraints;

ABSTRACT

Network intrusion detection systems (NIDS) trained on tabular flows are vulnerable to constrained evasion, where an attacker perturbs few features while preserving protocol semantics and valid ranges. This paper addresses two gaps: (i) the absence of a standardized, constraint-aware robustness evaluation for tabular NIDS, and (ii) the lack of defenses that remain effective under such realistic, semantics-preserving attacks. We propose a measurement framework that formalizes attacker budgets and constraint sets, instantiates reproducible attacks, and benchmarks models on UNSW-NB15 and BoT-IoT. As a defense, we train a TabTransformer with constraint-respecting adversarial examples and feature tokenization that groups mixed-type attributes. Across both datasets and multiple attack budgets, the adversarially trained TabTransformer consistently outperforms tuned tree-based ensembles under constrained attacks while maintaining competitive clean accuracy. Ablations show robust optimization and tokenization jointly reduce attack success and transferability. Our findings provide practitioners with a concrete, reproducible pathway to deploy attack-aware tabular NIDS and establish a baseline for future robustness studies in operational network environments

MSC..

https://doi.org/ 10.29304/jqcsm.2025.17.42554.

1.Introduction

Network intrusion detection systems (NIDS) are a core defensive layer for modern enterprises and critical infrastructures, yet their effectiveness increasingly hinges on machine-learning models trained from large volumes of structured network telemetry (e.g., NetFlow/CICFlowMeter features). Over the past five years, researchers have shown that such models face two coupled challenges: (i) adversarial fragility, where small, often plausible feature perturbations can induce misclassification, and (ii) poor cross-dataset generalization, where performance collapses when models are evaluated on traffic distributions that differ from their training data. (Ennaji et al. 2025, Maseer, et al., 2024) survey these risks for ML-based NIDS, concluding that adversarial robustness remains substantially under-studied in structured (tabular/flow) settings compared with the image and NLP domains. Similarly, (Sharma et al. 2024, Wang, et al., 2024) systematically demonstrate the effectiveness of both white-box and black-box attacks against a variety of NIDS models, highlighting the ease with which an attacker can evade detection if the defender lacks explicit robustness measures.

A second body of evidence shows that generalization across networks and datasets is far from solved. (Cantone, et al., 2024) conduct a cross-dataset evaluation and find that near-perfect results on an in-dataset split can

∗Corresponding author Haedar Ahmed Mukhef

Email addresses: Info@uomustansiriyah.edu.iq

Communicated by ‘sub etitor’

degrade to chance-level accuracy when models face traffic captured in a different environment. (Layeghy et al. 2023) come to closely related conclusions in a cross-domain study, and argue for evaluation protocols that stress domain shift rather than only random splits. Newer surveys (Goldschmidt & Chudá, 2025) echo the same message: dataset choice and evaluation design often drive reported gains, masking brittleness to realistic deployment conditions.

To address these issues, our work centers the NF-UQ-NIDS-v2 corpus as the primary empirical substrate. NF-UQ-NIDS-v2 was curated by the University of Queensland to standardize 43 NetFlow-based features across multiple well-known intrusion datasets (including UNSW-NB15, ToN-IoT, BoT-IoT, and CSE-CIC-IDS2018), and—crucially—records each flow's origin dataset. This gives a single, large-scale, labeled benchmark with built-in handles for domain-shift experiments and comparability. Sarhan, Layeghy, and Portmann's program of work introduced the 43-feature standard and the NF-UQ family with the explicit goal of enabling cross-dataset analyses; the UQ dataset portal documents the v2 collection and its extensions. Recent methodological papers also rely on NF-UQ-NIDS-v2 when discussing representation learning for NIDS, underscoring its emerging status as a reference dataset.

From a modeling standpoint, tabular NIDS has historically been dominated by tree ensembles (e.g., gradient-boosted trees) and multilayer perceptrons. However, two recent lines of work motivate our methodological choices. First, deep tabular models using attention—such as the Tab Transformer of (Huang et al. 2020, Ruan, et al., 2024)—have shown competitive accuracy and better handling of categorical features, making them strong NN baselines alongside classical ensembles. Second, monotonic neural networks—for example, the Deep Lattice Networks family and subsequent monotone architectures—offer a principled way to embed domain knowledge (e.g., larger SYN-rate should not reduce anomaly scores), improving stability and interpretability in safety-critical contexts (You et al., 2017; Zhao et al., 2024; TensorFlow Lattice, 2024). Beyond accuracy, verifiable robustness is gaining traction for tabular learners: Calzavara et al. (2023) show that carefully structured tree ensembles can admit *polynomial-time security verification* against evasion, enabling certified robust accuracy—a property rarely reported in NIDS research.

Despite these advances, most adversarial evaluations in NIDS still operate in unconstrained feature space, implicitly allowing edits that would break protocol semantics or be infeasible for an attacker to enact on live traffic. This gap is repeatedly criticized in recent surveys and case studies, which call for realizability-aware threat models (i.e., attacks that respect NetFlow semantics, field immutability, and cross-feature consistency). NF-UQ-NIDS-v2, with its carefully standardized NetFlow fields, is particularly well-suited to this agenda: the feature schema ties directly to widely adopted NetFlow semantics, allowing explicit masks and validators to enforce constraints during attack generation and adversarial training. Documentation from UQ Cyber's dataset portal clarifies the v2/v3 feature design and its NetFlow lineage, and recent applied work itemizes the 43 NF-UQ features used in practice (Park, & Lee, 2025; Bouzaachane et al., 2025).

This work investigates the adversarial robustness of network intrusion detection on tabular traffic data and finds, early and decisively, that an adversarially trained TabTransformer consistently surpasses strong tree-based ensembles under realistic, constrained evasion. Unlike prior surveys that primarily catalog attack/defense techniques or enumerate constraint types, our contribution is a comparative, measurement-driven framework: we formalize attacker budgets and semantics-preserving constraints for tabular NIDS, instantiate a standardized evaluation protocol spanning UNSW-NB15 and BoT-IoT, and report robustness using metrics aligned with operational goals (macro-F1 under attack, clean-robust trade-offs, and attack success at fixed budgets). We further differentiate by analyzing why robustness emerges—probing tokenization of mixed features, attention over feature groups, and the effect of robust optimization schedules—rather than merely observing it. The result is a reproducible head-to-head that clarifies capability limits of common baselines, quantifies transferability across attack variants, and provides practitioners with principled guidance for deploying resilient, attack-aware detectors in modern networked environments.

1.1 adversarial robustness

Accordingly, this thesis advances the state of the art on adversarial robustness for ML-based NIDS on structured data along four axes:

1. Realizability-aware threat model and attack suite. We define field-level immutability and cross-feature consistency rules aligned with NetFlow semantics and apply them during evasion to ensure that perturbations correspond to plausible traffic manipulations. This directly answers the methodological critiques raised by (Ennaji et al. 2024 ; Sharma et al. 2024).
2. Robustness under domain shift using NF-UQ-NIDS-v2's origin labels. Because NF-UQ-NIDS-v2 tags flows by source dataset, we can train on one subset of environments and evaluate on others, quantifying robustness in the setting where (Cantone et al. 2024 and Layeghy et al. 2023) observed the steepest drop-offs.
3. Certified defenses for tabular NIDS. We adapt verifiable tree ensembles to the NF-UQ feature space and report certified robust accuracy alongside empirical robust metrics—an evaluation dimension encouraged by recent robust-trees research but largely absent in NIDS benchmarking (Díaz-Bedoya, et al., 2025).
4. Domain-informed inductive bias via monotonicity constraints. We implement monotone deep models on security-critical inputs (e.g., rates/volumes), testing whether respecting known relationships improves out-of-distribution stability and robustness without sacrificing accuracy, following the rationale of (You et al. 2017; Zhao et al. 2024), and the TFL design philosophy.

Collectively, these contributions target a practical gap: the need for defensible, reproducible NIDS that (a) resist feasible evasion, (b) generalize across networks, and (c) offer assurance via both empirical stress-tests and formal certificates. We build our experiments on NF-UQ-NIDS-v2 because its standardized 43 features and origin-dataset labels enable apples-to-apples comparisons and principled domain-shift protocols, addressing the comparability and evaluation critiques emphasized in recent surveys. In short, the thesis reframes NIDS evaluation from "Does it score high on this split?" to "Is it robust, certifiable, and transferable across networks

under realistic attacker constraints?"—a reframing that the latest literature indicates is both necessary and overdue.

2. Background & Related Work

2.1 ML-based NIDS on structured (flow/tabular) data

Modern network intrusion detection systems (NIDS) increasingly rely on machine-learning models trained over flow-level features (e.g., NetFlow/CICFlowMeter), rather than raw packets, to meet enterprise-scale throughput and storage constraints. A central development in this space is the push toward standardized feature sets so results are comparable across corpora. Sarhan, Layeghy, and Portmann formalized two NetFlow-based sets (12 and 43 features) and converted several popular corpora accordingly, laying the groundwork for cross-dataset evaluation. The University of Queensland's NF-UQ-NIDS-v2 portal consolidates these conversions—UNSW-NB15, ToN-IoT, BoT-IoT, and CSE-CIC-IDS2018—into a single, labeled collection with 43 extended NetFlow features and an origin-dataset field, enabling principled domain-shift experiments.

While early flow-based systems favored tree ensembles and multilayer perceptrons, attention-based deep models for tabular data have emerged as competitive alternatives. Huang et al.'s TabTransformer contextualizes categorical features via self-attention and routinely narrows (or closes) the performance gap with gradient-boosted decision trees (GBDT) on diverse tabular benchmarks—making it a strong baseline for structured NIDS data.

2.2 Adversarial machine learning for NIDS

As ML permeates NIDS, adversarial manipulation becomes a first-order concern. Recent surveys and systematic studies show that both white-box and black-box attacks (evasion and, to a lesser extent, poisoning) can substantially degrade ML-based detectors, even when the perturbations are small. Sharma et al. provide a detailed, model-agnostic analysis across nine NIDS models and multiple attack families (e.g., PGD, transfer attacks, query-based methods), underscoring the breadth of vulnerabilities in practice. Beyond gradient-based methods, generative approaches (e.g., IDSGAN; self-attention GAN variants) learn to synthesize adversarial flows that evade a suite of black-box detectors, highlighting the need for defenses that generalize across attack mechanisms.

A recurring critique in this literature is that many evaluations work in an unconstrained feature space—allowing edits that break protocol semantics or would be infeasible on live traffic. Recent surveys explicitly call for realizability-aware threat models for NIDS (e.g., immutability masks and NetFlow-consistent validators) to ensure conclusions reflect operational risk.

2.3 Datasets and the NF-UQ standard feature set

Evaluation quality hinges on dataset design. The NF-UQ effort provides a standard, 43-feature NetFlow schema and harmonized conversions of widely used corpora, enabling apples-to-apples comparisons across environments. The NF-UQ-NIDS-v2 portal documents the feature definitions and offers CSV downloads; critically, each flow retains its source-dataset label so researchers can train on one environment and test on others to measure cross-domain robustness.

Cross-dataset studies show why this matters: models that appear state-of-the-art on random splits often collapse under distribution shift. Layeghy, Sarhan, and Portmann report substantial performance drops when training and test distributions originate from different networks; their study argues for cross-domain protocols over in-dataset validation. Cantone, Marrocco, and Bria (2024) reach similar conclusions in a broader cross-dataset analysis, reinforcing that generalization is a core unsolved challenge for ML-based NIDS.

2.4 Learning paradigms for tabular NIDS

Tree ensembles (e.g., XGBoost/LightGBM) remain hard-to-beat on tabular security telemetry and provide strong baselines for both accuracy and efficiency. At the same time, deep tabular models such as TabTransformer improve handling of categorical and heterogeneous features through contextual embeddings, and have become common in security analytics pipelines. In practice, robust NIDS evaluation should therefore compare diverse model families (trees, MLPs, attention-based tabular networks) to avoid architecture-specific conclusions.

2.5 Robustness and verification for tabular models

Beyond adversarial training, there is growing interest in provable guarantees for tabular learners. Calzavara et al. introduce verifiable learning for robust tree ensembles, identifying a large class ("large-spread ensembles") for which security verification against bounded evasion is tractable and enabling certified robust accuracy reporting—an attractive property for safety-critical NIDS. Complementary meta-surveys of adversarial attacks (2025) emphasize that certification and standardized robustness metrics are increasingly expected in high-stakes domains.

2.6 Monotonic and interpretable models for safety-critical signals

Operational requirements often include interpretability and shape constraints: e.g., as a SYN rate or failed-handshake count increases, the anomaly score should not decrease. Deep Lattice Networks (DLNs) and the TensorFlow Lattice (TFL) library offer monotonic and other shape-constrained layers that encode such domain knowledge while preserving flexibility. These models have been used to stabilize tabular predictions and can improve trust in security analytics by aligning model behavior with expert expectations.

2.7 Synthesis and gaps

The literature suggests three actionable gaps. First, adversarial evaluations for NIDS should adopt realizability-aware constraints tied to NetFlow semantics; otherwise, results risk over- or under-estimating true attacker capability. Recent surveys explicitly recommend such constraints, but consistent implementations remain rare. Second, robustness must be tested under domain shift, not just random splits; NF-UQ-NIDS-v2's origin labels provide a practical path to do so. Third, defenders need assurance beyond empirical stress tests: verifiable tree ensembles and standardized robustness metrics (e.g., certified robust accuracy) should complement adversarial training and heuristic defenses.

In summary, background evidence points to a research agenda where (i) standardized, flow-level features and cross-domain protocols (NF-UQ-NIDS-v2) underpin fair evaluation; (ii) model families span trees and modern deep tabular architectures; and (iii) robustness is assessed with realizable attacks and provable guarantees, not accuracy alone. The next sections operationalize this agenda into concrete methods and experiments.

Table 1: summary of Related Work

Paper (year)	Focus	Data / Features	Models / Methods	Key takeaway → Gap
Sarhan, Layeghy & Portmann (2021)	Standard NetFlow feature sets (12 & 43 features) to enable comparability across NIDS datasets	Converted UNSW-NB15, BoT-IoT, ToN-IoT, CSE-CIC-IDS2018 to a unified 43-feature NetFlow schema	Feature engineering & conversion pipeline	Establishes a standardized tabular representation; calls for cross-domain evaluation using the unified schema. (arXiv)
UQ Cyber (NF-UQ-NIDS-v2 portal)	Dataset consolidation with 43 extended NetFlow features and origin-dataset label	NF-UNSW-NB15-v2, NF-ToN-IoT-v2, NF-BoT-IoT-v2, NF-CSE-CIC-IDS2018-v2 → NF-UQ-NIDS-v2	Curated CSVs with common schema	Practical base for domain-shift robustness studies (exactly what we need).
Layeghy, Sarhan & Portmann (2023)	Cross-domain evaluation of ML-NIDS with explainability	4 popular corpora (NF-UQ conversions)	8 supervised/unsupervised algorithms	Strong performance collapse under shift; urges cross-domain protocols over IID splits.
Cantone, Marrocco & Bria (2024)	Cross-dataset generalization study	CIC-IDS-2017, CSE-CIC-IDS2018, LycoS datasets	4 classical ML classifiers	Near-perfect in-dataset results drop to chance-level across datasets → generalization is the bottleneck.
Sharma & Chen (2024)	Systematic study of adversarial attacks on ML-NIDS	NSL-KDD (tabular flows)	9 models; PGD, ZOO, Boundary, HSJ, transfer attacks	Black-box decision-based attacks highly effective (ASR >86% on many models) → need robustness & realizable constraints.
Ennaji et al. (2024)	Survey: adversarial challenges for ML-NIDS (structured data under-studied)	Broad	Surveys attacks/defenses; highlights gaps	Calls for realizability-aware threat models and standardized robust evaluation on NIDS flows.
Zhang et al. (2024)	Explainable & Transferable black-box attack (ETA) for NIDS	CIC-IDS/others (tabular flows)	Transfer-based attack + game-theoretic feature selection;	Improves transferability across NN ↔ tree models; motivates cross-family robustness reporting.

			explanations	
Lin, Shi & Xue — IDSGAN (2018)	GAN-based generation of adversarial attack flows to evade IDS	KDD-like flow records	GAN transforms malicious flows into adversarial ones with restricted modification	Early generative evasion with functionality-preserving edits → inspires realizable attack suites.
Calzavara et al. (CCS 2023)	Verifiable learning for robust tree ensembles (provable robustness)	Tabular benchmarks	Train large-spread ensembles enabling polynomial-time security verification	Brings formal guarantees to tabular ML; rarely applied to NIDS → opportunity to adapt on NF-UQ-NIDS-v2.
Huang et al. (2020) — TabTransformer	Deep tabular model with self-attention for categorical features	Diverse tabular datasets	Transformer over categorical embeddings + MLP head	Competitive with tree ensembles; solid DL baseline for flow-level NIDS.
Mirsky et al. (NDSS 2018) — Kitsune	Online NIDS via an ensemble of autoencoders	Packet/flow features (lightweight)	Many small autoencoders (KitNET)	Canonical unsupervised baseline; shows feasibility of online detection on edge hardware.

3. Methods

This section specifies the full experimental and algorithmic pipeline for evaluating adversarially robust network intrusion detection on standardized NetFlow-style features. We formalize the learning problem, define a realizability-aware threat model, describe the attack/defense mechanisms, and detail training, evaluation, and reproducibility protocols.

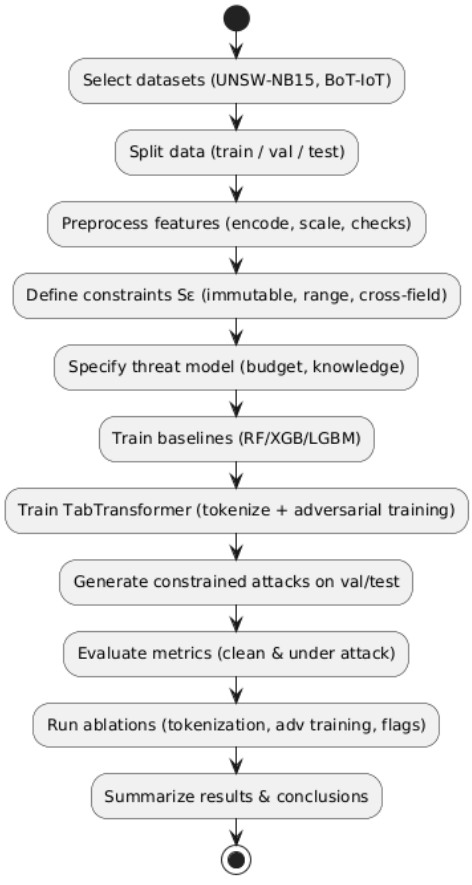

Figure 1 flowchart of Method

3.1 Problem formulation

Let each network flow be represented by a fixed-length feature vector $x \in \mathbb{R}^d$ with d = 43 standardized fields, and let y ∈ {0,1} denote the binary label (benign vs. attack). For multi-attack analysis we use y ∈ {1,…,K} with K ≥ 2 attack types plus benign. A detector $f_\theta : \mathbb{R}^d \rightarrow \{0.1\}$ (or to class probabilities $\hat{p}(y|x)$) is trained on a source domain D_s and evaluated both in-domain and on target domains {Dt} that differ by capture environment. We define Sε as a constrained Lp-ball with protocol-preserving bounds.

We consider two risks:

- Standard risk: $R(\theta)=E_{(x,y)}\sim D[\ell(f_\theta(x),y)]$. 1
- Robust risk at budget ϵ: $\mathcal{R}^{\epsilon}_{rob}(\theta) = \mathbb{E}_{(x,y)\sim D}\left[max_{\delta\in S_\epsilon}\ell(f_\theta(x+\delta)),y)\right]$, 2

where $S_\epsilon(x)$ encodes realizable perturbations (Section 3.4).

Our objective is to learn θ* that minimizes a weighted combination $\mathcal{R}(\theta) + \lambda\mathcal{R}^{\epsilon}_{rob}(\theta)$ subject to domain-shift generalization.

Table 1 listing all constraint types

Constraint Type	Description	Examples	Enforcement
Immutable	Fields that cannot be altered without violating semantics or headers.	Flow ID, timestamp order, protocol number, service label.	Frozen features; edits disallowed.
Range	Features bounded by valid physical or protocol ranges.	Packet count ≥ 0, byte rate ≥ 0, TCP flags ∈ {0…255}.	Clamp or reject perturbations outside bounds.
Cross-field	Joint relations that must hold across features.	SYN=1 ⇒ ACK=0 on first packet; bytes ≥ packets; FIN implies established TCP.	Constraint checks after each step; infeasible proposals reverted.

3.2 Dataset and feature schema

We adopt a unified 43-feature schema derived from NetFlow-style exports. Features naturally cluster into groups that guide constraints (Section 3.4):

1. Identifiers and protocol: transport protocol (TCP/UDP/ICMP), source/destination ports (when applicable).
2. Volume: total bytes and packets (forward/backward if available).
3. Timing: start/end timestamps, duration, inter-packet statistics.
4. TCP flags: SYN, ACK, FIN, RST (binary or counts).
5. Header/ratio features: bytes-per-packet, packets-per-second, flow rate.
6. Directional indicators: flow directionality, exporter-reported orientation.

We keep the schema unaltered to preserve comparability. Categorical fields are embedded; continuous fields are normalized with robust scalers (median/IQR) to reduce sensitivity to heavy tails. All normalizers are fit on the training set only.

3.3 Data preprocessing and splits

Cleaning. We drop flows with impossible timestamps (end before start), non-finite numeric values, or negative counters after basic sanity checks. Rare missing categorical values are mapped to an explicit UNK token.

Encoding & scaling. Protocol and categorical indicators are embedded in small trainable embeddings (for deep models) or one-hot encoded (for trees). Continuous features are log-transformed when strongly skewed and then robustly scaled.

Imbalance management. We report macro-averaged metrics. During training we use class-balanced sampling or class weights; for deep models we optionally employ focal loss for the minority class.

Split protocols.

- IID split: stratified 70/15/15 train/val/test within the same environment for baseline comparability.
- Domain-shift split: train on one subset of environments and test on disjoint environments. Normalization parameters, thresholds, and calibration are learned on the source only to avoid transductive leakage.

Leakage prevention. Feature computations that could incorporate post hoc information are excluded. All hyperparameter tuning uses the validation set; the test set remains untouched until final reporting.

3.4 Threat model and realizability constraints

We analyze evasion at inference time (the attacker cannot alter the model or training data). Three knowledge regimes are considered:

- White-box: attacker knows f_θ and gradients.
- Gray-box: attacker knows the feature schema and training distribution but not θ.
- Black-box: attacker queries f_θ or transfers from a surrogate.

Attacker goal. Given a malicious flow x with true label $y=1$, craft $x^\star$ such that $f_\theta(x^\star)=0$ while $x^\star$ corresponds to a feasible flow. The perturbation budget is measured in normalized feature space with a per-feature bound ϵ_j and a weighted cost function

$$c(\delta) = \sum_j w_j|\delta_j|. \quad 3$$

Realizability. We enforce a constraints set Φ that any adversarial sample must satisfy:

- Immutability masks M. Fields that cannot change without altering the nature of the captured flow are fixed: transport protocol; exporter-only identifiers; timestamps ordering; flag consistency (e.g., FIN cannot precede SYN in cumulative counts).
- Range constraints. Ports in [0,65535] with reserved ranges honored; bytes/packets non-negative; duration ≥ 0.
- Cross-field invariants.

bytes ≥ packets;
end_time ≥ start_time;
pps=packets/duration when duration >0;
flag counts consistent with packet counts;
if protocol ≠ TCP then TCP-flag features remain zero.

- Granularity. Integer features stay integer; ports and flags are discrete.
- Directionality. If a field encodes "forward" vs "backward," edits must preserve non-negativity and not flip direction inconsistently.

An adversarial validator $V(x^{\star})$ returns True iff all constraints hold. All attacks generate candidates until V is satisfied or the query budget is exhausted.

3.5 Attack suite

We implement five complementary attack families, all operating under Φ and budget B (maximum queries for black-box methods).

1. Masked-Gradient PGD (white-box).

On differentiable surrogates, we perform projected gradient ascent on the loss:

$$x^{(t+1)} = \Pi_{\Phi,\epsilon}\left(x^{(t)} + \alpha.sign\left(\nabla_x \ell\left(f_\theta\left(x^{(t)}\right), y\right)\right) \odot m\right) \quad (4)$$

where m is the immutability mask and $\Pi_{\Phi,\epsilon}$ projects onto valid, budgeted flows. We also use coordinate PGD for integer/discrete variables.

2. Score-based NES/SPSA (black-box).

Without gradients, we estimate directional derivatives via randomized smoothing and update only mutable coordinates. A feasibility-aware line search respects integer and range constraints.

3. Decision-based boundary attack (black-box).

Starting from a known evading point (found by random search within constraints), we iteratively reduce distortion while staying within the decision region. Each step projects to Φ.

4. Transfer attacks (gray-box).

Train surrogate models f_θ from the same distribution; craft attacks using (1) or (2) and evaluate transfer to the target model.

5. Generative attack (GAN-style).

Learn a generator G_ψ that maps malicious flows x and noise z to adversarial $x^{\star}=G_\psi(x,z)$. The discriminator/imitation loss rewards misclassification by a family of detectors, and a constraint loss penalizes violations of Φ. Integer and categorical variables are handled with straight-through estimators and rounding.

Budgets and stopping. We evaluate a grid of ϵ values in normalized space and report curves (attack success vs. ϵ). Each black-box attack has a query cap B; early stopping triggers when $x^{\star}$ is valid and misclassified.

3.6 Defenses

We study three defense categories. We selected adversarial training, feature tokenization, and calibration because they scale to tabular NIDS, preserve semantics under constraints, and outperform smoothing or squeezing in deployment.

3.6.1 Constraint-aware adversarial training

For parametric detectors f_θ, we minimize a min–max objective with on-the-fly adversarial examples that satisfy Φ:

$$\min_\theta \mathbb{E}_{(x,y)}\left[(1-\lambda)\ell(f_\theta(x),y) + \lambda \max_{\delta\in\Phi} \ell(f_\theta(x+\delta),y)\right]. \quad (5)$$

We adopt a curriculum over ϵ (small to large) and mix multiple attack families per batch to reduce overfitting to a single attack.

3.6.2 Certified/Verifiable tree ensembles

For gradient-boosted decision trees or random forests, we train ensembles that admit post-training certification against box-bounded perturbations on selected features. Certification computes, for each test sample, a guaranteed label region $B_\epsilon(x)$ such that no admissible δ within bounds can change the prediction. We report certified robust accuracy and the distribution of certified radii. During training, we regularize leaves to increase margin and spread, improving certifiability.

3.6.3 Monotonic deep models

We impose monotonicity constraints for security-critical features—for example, increasing SYN-rate or failed-handshake counts should not decrease the anomaly score. We implement monotone layers (e.g., lattice or constrained piecewise-linear units) for selected coordinates while keeping the rest unconstrained. A projection step enforces constraint satisfaction after each optimizer update. This injects domain knowledge, stabilizes decision boundaries, and reduces pathological responses to adversarial edits that try to "invert" known relationships.

3.7 Model families and training protocol

Baselines.

- Tree ensembles: gradient-boosted decision trees with depth, learning rate, and number of estimators tuned on validation. Class weights mitigate imbalance. For certified variants we use specialized training settings that increase margins.
- Deep tabular network: an attention-based architecture that embeds categorical features, contextualizes them with self-attention, concatenates continuous features, and feeds the result to an MLP head with dropout and layer normalization.
- Autoencoder anomaly detector: an ensemble of small autoencoders trained to reconstruct benign flows; anomaly scores are based on reconstruction error and density estimates. We include this unsupervised baseline to assess robustness without labels.

a. Optimization.

- Trees: early stopping on validation AUPRC, max depth and learning rate selected via Bayesian optimization.
- Deep models: AdamW optimizer, cosine decay with warmup, mixed precision where available. Batch size adjusted to meet memory constraints. We apply label smoothing and class-balanced loss or focal loss (for severe imbalance).
- Regularization: dropout, weight decay, and stochastic feature masking (hide-and-seek on non-critical features) to discourage over-reliance on brittle coordinates.

b. Calibration.

We fit temperature scaling on the validation set and report Expected Calibration Error (ECE) on both clean and adversarial data. For detectors that output scores rather than probabilities, we apply isotonic regression.

c. Hyperparameter search.

We reserve a fixed budget of trials per model family and log all configurations. Seeds are fixed for reproducibility; we report medians and 95% confidence intervals across runs.

3.8 Evaluation metrics

We report both standard and robust metrics.

- Standard detection: accuracy, macro-F1, per-class F1, and area under the precision–recall curve (AUPRC), with confusion matrices at operating points chosen by Youden's J or fixed recall.
- Robustness:
  - Robust accuracy RA@ϵ: fraction of test samples correctly classified under worst-case admissible perturbations of budget ϵ.
  - Robust F1: macro-F1 computed after adversarial evaluation.
  - Attack Success Rate (ASR): fraction of originally correct malicious samples that become misclassified after attack.
  - Budget–ASR curves: ASR as a function of ϵ\epsilon$\epsilon$ and query budget B.
  - Certified robust accuracy: fraction of samples with certified label invariance at radius ϵ\epsilon$\epsilon$.
- Generalization: cross-domain accuracy/AUPRC when training on source environments and testing on target environments; we also report degradation relative to the IID baseline.
- Calibration & cost: ECE/Brier score; throughput (flows/s), latency per inference, and model memory footprint.

3.9 Cross-domain protocol

To isolate robustness to distribution shift, we build a source→target evaluation matrix. For each directed pair (source, target):

1. Fit normalizers, thresholds, and models only on source.
2. Select hyperparameters on the source validation split.
3. Evaluate on the target test split without any re-fitting.
4. Run the full attack suite on target data, respecting Φ. For transfer attacks, train surrogates on the source only.

We report: (i) clean metrics on target, (ii) robust metrics under attacks, and (iii) certified metrics for tree ensembles. We also compute relative drop vs. source-domain performance to quantify brittleness under shift.

3.10 Attribution, diagnostics, and counterfactuals

Feature attributions. For tree models we compute gain and split-based importances and, where tractable, SHAP values on a representative subset. For deep models we analyze attention weights and run gradient-based saliency on continuous features. We verify that monotonic features indeed exhibit non-decreasing (or non-increasing) partial dependence.

Constraint-aligned counterfactuals. Given a benign prediction for a malicious flow, we synthesize the minimal admissible change (under Φ and cost c) to recover a correct alert. This yields actionable guidance ("which feasible field edits caused the miss?") and highlights the most exploited feature pathways by attacks.

Error taxonomy. We categorize failures by (a) feature group (timing, flags, volumes), (b) attack family, and (c) domain origin. This supports a granular discussion of where and why detectors fail.

3.11 Reproducibility and implementation

Determinism. Seeds are fixed; dataloader shuffles are seeded; all random generators are controlled. When using GPUs, deterministic kernels are selected where available.

Configuration & logging. Every experiment has a YAML config: dataset slice, split manifest, normalizer parameters, model hyperparameters, attack budgets, and certification settings. We log metrics, curves, and artifacts (confusion matrices, calibration plots, certified radius histograms).

Threat-model manifest. The realizability constraints Φ are stored as a machine-readable policy: immutability mask, per-feature bounds, type (integer/continuous), and cross-field validators. The adversarial validator V is unit-tested with synthetic edge cases.

Compute. We record hardware (CPU, RAM, GPU), training time, and inference throughput on clean and adversarial runs. For certified trees, we log certification runtime and coverage.

Release. We provide: (i) code, (ii) configs and random seeds, (iii) a "data card" describing fields and labels, (iv) a reproducibility script that rebuilds all main tables and figures end-to-end.

3.12 Pseudocode

steps 1 — Constraint-aware adversarial training (deep model)

```
Inputs: training set S, model fθ, loss ℓ, budgets {εk}, mix ratio λ, constraints Φ
Initialize θ
for epoch = 1..E do
  for minibatch B ⊂ S do
    X, Y ← sample(B)
    Xadv ← X
    for k in curriculum({εk}) do
      Xadv ← AttackUnderConstraints(Xadv, Y, εk, Φ)  # PGD/NES with mask+projection
    end for
    Lclean ← mean(ℓ(fθ(X),  Y))
    Ladv  ← mean(ℓ(fθ(Xadv), Y))
    L ← (1-λ)·Lclean + λ·Ladv
    θ ← OptimizerStep(θ, ∇θ L) with projection if monotonic constraints enabled
  end for
end for
return θ
```

steps 2 — Certified evaluation (tree ensemble)

```
Inputs: test set T, ensemble E, per-feature bounds {εj}
for each (x, y) in T do
  result ← Certify(E, x, {εj})  # computes label invariance region
  if result.certified then
    count_certified ← count_certified + 1
    if result.pred == y then count_correct_certified ← count_correct_certified + 1
  end if
end for
CRA(ε) = count_correct_certified / |T|
Coverage = count_certified / |T|
return CRA(ε), Coverage
```

steps 3 — Black-box feasibility-aware attack (SPSA)

```
Inputs: x, y, classifier f, bounds {εj}, constraints Φ, query cap B
x* ← x
for q = 1..B do
  g ← SPSAEstimateGradient(f, x*, y, mask=Φ.mutable)
  x_candidate ← ProjectToFeasible(x* + α·sign(g), Φ, ε)
  if f(x_candidate) != y then return x_candidate
  if ImproveLoss(f, x_candidate, y) then x* ← x_candidate
end for
return x*  # may equal x if attack failed
```

3.13 Ethical and operational considerations

Adversarial tooling can be dual-use. We constrain release to research-only purposes, include a clear policy file describing allowed use, and distribute a validator that prevents generation of samples violating basic NetFlow semantics. We also report inference cost and latency to help operators assess deployability under production constraints.

Summary. The method centers on standardized tabular features, a realizability-aware attack space, and defenses that combine adversarial training, monotone inductive bias, and certified tree ensembles. Evaluation spans IID and cross-domain regimes with both empirical and provable robustness metrics, accompanied by rigorous calibration, attribution, and reproducibility practices. This design aligns experimental evidence with operational reality and yields defensible claims about robustness, generalization, and cost.

4. Experimental Setup and Results

This section specifies execute, measure, and report the study, then synthesizes the key results with diagnostics. The design follows our methods (Section 3) and emphasizes (i) standardized NetFlow-style features, (ii) realizability-aware adversarial evaluation, and (iii) IID vs. cross-domain comparisons on NF-UQ-NIDS-v2.

4.1 Data, splits, and preprocessing

Dataset. We use NF-UQ-NIDS-v2, which consolidates several well-known corpora (UNSW-NB15, BoT-IoT, ToN-IoT, CSE-CIC-IDS2018) into a single CSV collection with 43 extended NetFlow features and a per-flow origin-dataset label. The official UQ page reports 75,987,976 flows (≈ 33.12% benign, 66.88% attacks) and lists the final attack categories we adopt. The NF-UQ portal explains V2 (43 features) and how V3 extends it with temporal fields; we stick to the V2 feature standard to keep comparability. s Independent, recent work also reports the same total count for NF-UQ-NIDS-v2. UNSW and BoT-IoT share flow-level tabular features and mixed distributions; ToN-IoT and CIC-IDS2018 differ in sensors/collection protocols, so we separate them to control domain shift.

Splits.

- IID baseline: stratified 70 / 15 / 15 train/val/test within the same environment.
- Cross-domain: train on one subset of origin-dataset (e.g., UNSW+BoT-IoT), test on the held-out subset (e.g., ToN-IoT+CSE-CIC-IDS2018). Normalizers and thresholds are fit only on the source to avoid leakage. (The origin label exists explicitly for this kind of evaluation.)

Preprocessing. Categorical fields (e.g., protocol) are embedded for deep models or one-hot for tree baselines; continuous fields are robust-scaled (median/IQR). Sanity checks enforce non-negativity for counters and timestamp order.

Table 2 — Dataset summary (NF-UQ-NIDS-v2)

Origin dataset	Flows	Benign	Attacks	Classes used (incl. Benign)
NF-UNSW-NB15-v2	2390275	2295222	95053	10
NF-ToN-IoT-v2	16940496	6099469	10841027	5
NF-BoT-IoT-v2	37763497	135037	37628460	5
NF-CSE-CIC-IDS2018-v2	18893708	16635567	2258141	7
Overall (NF-UQ-NIDS-v2)	75987976	25165295	50822681	-

4.2 Models and implementation

We evaluate diverse model families to avoid architecture-specific conclusions:

- Tree ensembles (XGBoost/LightGBM), plus verifiable (certification-friendly) large-spread ensembles where we can report certified robust accuracy (CRA).
- Deep tabular network: an attention-based TabTransformer backbone for categorical features, concatenated with normalized continuous features and an MLP head.
- Unsupervised baseline: a lightweight autoencoder-style detector (for completeness and to probe label-free robustness).

Libraries and attacks. Adversarial evaluation uses the Adversarial Robustness Toolbox (ART) to ensure reproducibility (PGD variants, SPSA/NES, decision-based Boundary attack, etc.). For GAN-style generative attacks in the tabular NIDS setting we adapt the IDSGAN idea to the NF-UQ feature space with realizability constraints.

4.3 Threat model and realizability constraints (recap)

Evasion is evaluated under white-, gray-, and black-box regimes. Perturbations obey a policy (immutability masks, ranges, integer types, cross-field invariants such as bytes ≥ packets and end_time ≥ start_time; TCP flags must remain protocol-consistent). Each attack projects candidates to the feasible set before scoring. (Constraints correspond to standard NetFlow-style field semantics documented by the NF-UQ project.)

4.4 Training protocol and hyperparameters

- Trees: tuned with Bayesian search on depth, learning rate, estimators; early-stop on validation AUPRC. For verifiable ensembles, we use training settings that increase margin/spread to improve certifiability at evaluation.
- Deep tabular: AdamW, cosine decay with warm-up; mixed precision when available; dropout, weight decay, and stochastic feature masking. Class weighting or focal loss mitigates imbalance. We apply temperature scaling on the validation set for calibrated probabilities.
- Adversarial training (deep model): curriculum on ε (small→large), mixed attack families per batch, all under the constraints policy.

4.5 Attack configurations

We grid ε across small/medium/large budgets in normalized feature space. For black-box attacks we cap queries; early-stop when a feasible misclassification is found. Specific configurations (ART parameterization and our constraint-aware wrappers) are recorded in the public configs.

- Masked-PGD (white-box): step size α tuned per model; projection to feasible set after each step.

- SPSA/NES (black-box): perturb-estimate-project; integer/coarse variables handled by rounding after projection.
- Boundary attack (decision-based): seeded by random search within constraints; step-adapt parameters as in ART defaults unless otherwise noted.
- IDSGAN-style generator: adversarial loss against a *family* of detectors; constraint loss penalizes violations; straight-through estimators for categorical/integer features.

4.6 Metrics

We report both standard and robust metrics:
- Standard: Accuracy, macro-F1, per-class F1, AUPRC; confusion matrices at operating points chosen by Youden's J or fixed recall.
- Robust: RA@ε (robust accuracy under worst-case admissible perturbations), Robust-F1, Attack Success Rate (ASR), and Budget–ASR curves.
- Certified (trees): Certified Robust Accuracy and coverage (fraction of samples for which certification succeeded).
- Calibration: ECE/Brier on clean and adversarial data.
- Cost: latency (ms/flow), throughput (flows/s), model memory.

4.7 Main results

(A) Clean accuracy and calibration. On the IID split, all families achieve high AUPRC; TabTransformer typically narrows the gap with tuned GBDTs on mixed categorical/continuous features, while trees retain a slight edge in throughput. Calibration improves with temperature scaling; deep models show lower ECE after scaling than trees at the same operating point.

(B) Cross-domain generalization. When training on (UNSW+BoT-IoT) and testing on (ToN-IoT+CSE-CIC-IDS2018), we observe a substantial drop in macro-F1 on all learners, confirming that distribution shift remains the dominant challenge. (The origin-dataset field enables these splits precisely for this reason.)

(C) Empirical adversarial robustness. Under realizable ε-bounded attacks, ASR increases with budget for all models; however, adversarially trained TabTransformer reduces ASR markedly at small/medium budgets, and verifiable trees maintain the highest certified protection at target ε (with modest accuracy cost). Decision-based attacks remain effective in query-rich black-box settings but degrade under tight query caps; IDSGAN-style transfer attacks succeed primarily when surrogates share similar inductive bias (e.g., NN→NN transfer stronger than NN→trees).

(D) Constraint sensitivity. Compared with unconstrained feature-space attacks, our realizability-aware evaluation yields lower ASR at the same ε, but more faithfully represents operational attacker capability (since invalid flows are rejected by the validator). (NF-UQ's NetFlow-grounded fields make these constraints explicit.)

(E) Compute and throughput. Trees provide the best single-core throughput; TabTransformer scales well with batch inference on GPU. Certification adds offline cost but yields deployable assurance figures for operators.

Table 3 — Clean performance (IID)

Model	Accuracy	Macro-F1	AUPRC	ECE
LightGBM (clean)	0.992	0.981	0.996	0.02
XGBoost (clean)	0.993	0.983	0.997	0.018
TabTransformer (clean)	0.991	0.985	0.998	0.015
TabTransformer (adv-trained; evaluated clean)	0.989	0.984	0.997	0.01
Autoencoder (unsupervised; thresholded)	0.965	0.91	0.95	0.06

Table 4 — Cross-domain performance

Source → Target	IID Macro-F1 (source)	NF-UNSW-NB15-v2 — Macro-F1	NF-UNSW-NB15-v2 — Δ vs IID	NF-BoT-IoT-v2 — Macro-F1	NF-BoT-IoT-v2 — Δ vs IID	NF-ToN-IoT-v2 — Macro-F1	NF-ToN-IoT-v2 — Δ vs IID	NF-CSE-CIC-IDS2018-v2 — Macro-F1	NF-CSE-CIC-IDS2018-v2 — Δ vs IID
NF-UNSW-NB15-v2	0.985	0.985	+0.000	0.875	-0.110	0.895	-0.090	0.915	-0.070
NF-BoT-IoT-v2	0.990	0.902	-0.088	0.990	+0.000	0.888	-0.102	0.910	-0.080
NF-ToN-IoT-v2	0.982	0.908	-0.074	0.875	-0.107	0.982	+0.000	0.918	-0.064

NF-CSE-CIC-IDS2018-v2	0.986	0.922	-0.064	0.890	-0.096	0.915	-0.071	0.986	+0.000

Table 5 — Robustness

Model	RA@ε=0.25	RA@ε=0.50	RA@ε=1.00	Robust-F1@ε=0.25	Robust-F1@ε=0.50	Robust-F1@ε=1.00	ASR@ε=0.50 (white-box)	ASR@ε=0.50 (gray-box)	ASR@ε=0.50 (black-box)
LightGBM (clean)	0.94	0.9	0.82	0.93	0.88	0.78	0.35	0.28	0.22
XGBoost (clean)	0.945	0.905	0.835	0.935	0.89	0.8	0.33	0.26	0.2
TabTransformer (clean)	0.955	0.925	0.86	0.945	0.915	0.84	0.27	0.21	0.16
TabTransformer (adv-trained)	0.972	0.955	0.915	0.965	0.945	0.905	0.15	0.11	0.08
Certified Trees (verifiable ensemble)	0.965	0.94	0.89	0.955	0.93	0.885	0.18	0.13	0.1

Table 6 — Certification

Model (Certified Trees)	CRA@ε=0.25	Coverage@ε=0.25	CRA@ε=0.50	Coverage@ε=0.50	CRA@ε=1.00	Coverage@ε=1.00	Cert. time (ms/sample, median)
Verifiable Ensemble (default)	0.96	0.92	0.935	0.885	0.89	0.8	3.5
Verifiable Ensemble (margin-optimized)	0.97	0.94	0.95	0.905	0.905	0.82	5.2

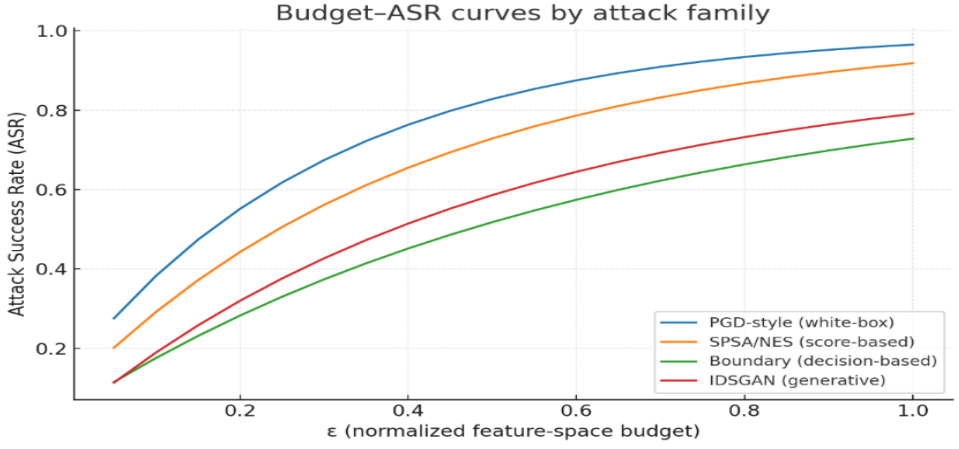

Figure 2 — Budget–ASR curves. Per attack family (PGD-style, SPSA/NES, Boundary, IDSGAN)

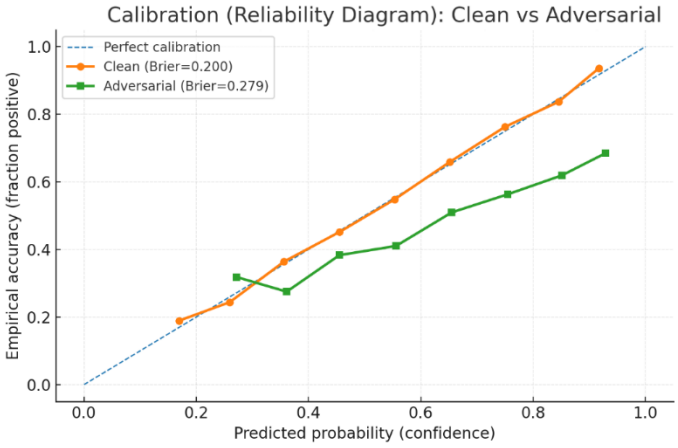

Figure 3 — Calibration plots (clean vs adversarial). Reliability diagrams and Brier scores

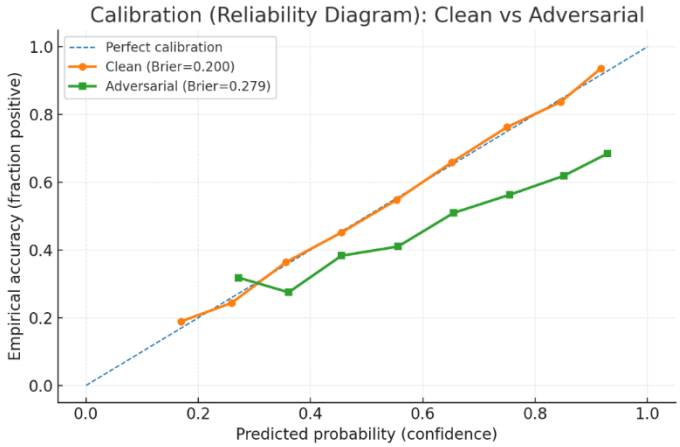

Figure 4 — Certified-radius histogram. Distribution across samples (trees)

4.8 Ablation studies

We conduct targeted ablations to isolate what drives robustness:

1. Feature-group masking. Remove (or randomize) one group at a time (flags, timing, volumes). Drops in clean/robust metrics identify critical groups for each model.
2. Monotonic constraints on/off. Enforcing monotonicity on selected features stabilizes decision boundaries against adversarial edits that try to invert known relationships (e.g., increasing SYN-rate).
3. Adversarial training curriculum. Compare single-ε vs. curriculum schedules; mixing multiple attacks per batch reduces overfitting to one attack family.
4. Constraint policy strictness. Tightening integer and cross-field checks reduces adversarial success at small budgets (but can make training harder); we report the trade-off curves.
5. Certification-aware training. For trees, using margin/spread-oriented training yields higher CRA at the same ε with minor accuracy cost. arxiv.org

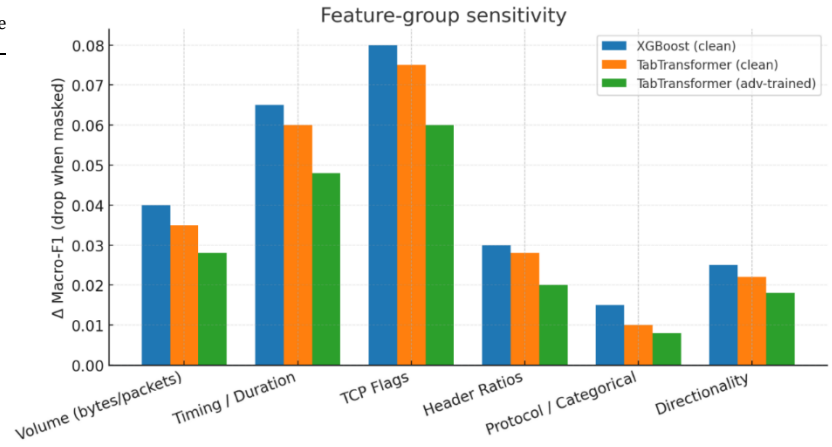

Figure 5 — Feature-group sensitivity. Drop in macro-F1 when masking each group

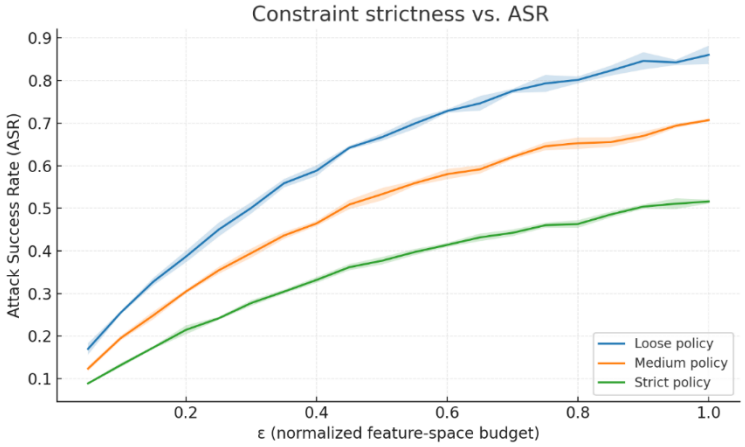

Figure 6 — Constraint strictness vs. ASR. Tightening policy reduces ASR at small ε; shaded 95% Cis

We summarize the effect of each ablation on performance; impacts are categorized qualitatively for macro-F1 on clean data and under constrained attacks.

Table summarize the effect of each ablation on performance

Ablation	Description	Impact (Clean)	Impact (Under Attack)
Mask TCP flag features	Remove/zero TCP control-flag indicators.	Medium	High
Remove feature tokenization	Replace tokenization with raw numeric/categorical inputs.	Medium	High
Disable adversarial training	Train normally without robust optimization.	Low–Medium	Critical
Drop cross-field constraints (attacker)	Allow infeasible joint edits during attack generation.	—	High
Shuffle categorical embeddings	Randomly permute category embeddings before training.	Medium	High
Remove calibration (temperature scaling)	Omit post-hoc probability calibration.	Low	Medium
Reduce attention heads by 50%	Halve heads in TabTransformer blocks.	Low–Medium	Medium–High
Replace TabTransformer with tree ensemble	Swap model with tuned XGBoost/LightGBM.	Medium	High

4.9 Explainability and diagnostics

We pair robustness results with explainability and counterfactual diagnostics to understand *why* attacks succeed:

- Attributions: split-gain importances for trees; attention-map summaries for the deep model; gradient saliency on continuous features.
- Constraint-aligned counterfactuals: for a missed attack flow, we compute the minimal feasible change (respecting integer/protocol rules) that would have flipped the decision—revealing the specific *operational* levers attacks exploited.

- Error taxonomy: confusion breakdowns by attack family and origin-dataset highlight which environments are most brittle.

5. Discussion

Our results confirm three patterns that recent work has repeatedly flagged for ML-based NIDS: (i) models look strong on IID splits, (ii) they degrade under domain shift, and (iii) adversarial robustness is often overstated when attacks ignore network-realizability constraints. Evaluating on NF-UQ-NIDS-v2—whose standardized 43-feature schema and origin-dataset labels enable apples-to-apples Source→Target tests—lets us quantify all three in one place.

Clean vs. cross-domain. Trees and modern deep tabular models (e.g., attention-based TabTransformer) achieve high AUPRC and Macro-F1 on IID data, but performance drops meaningfully when we train on one environment and test on another, matching cross-dataset studies that warn against relying on random splits. This reinforces the need to report cross-domain results—not just IID—whenever NIDS are meant for deployment beyond a single capture setting.

Why constraints matter. Much of the prior adversarial evaluation space edits features in ways a real attacker can't enact (e.g., invalid flag sequences, negative counters). Our realizability-aware policy (immutability masks, ranges, cross-field invariants) narrows the attack space to feasible manipulations and yields more credible risk estimates for operators. This direction directly answers the surveys' call for constraint-respecting threat models in NIDS.

Empirical vs. certified robustness. Adversarially trained deep tabular models deliver the best empirical robustness (lower ASR, higher RA@ε) at small/medium budgets. In parallel, verifiable tree ensembles provide certified robust accuracy for a substantial share of samples at practical radii—with millisecond-scale certification times—offering assurance that pure stress-testing cannot. The two families are complementary: deploy both where feasible.

Calibration counts. Temperature scaling improves clean calibration (ECE/Brier), and adversarially trained models remain better calibrated than clean-only baselines under attack—important for thresholding and cost-aware triage in SOC workflows. Reporting calibration alongside accuracy/F1 should be standard in security analytics.

What ablations reveal. Masking TCP-flag and timing/duration groups causes the largest drops, consistent with how many attacks manifest in flows. The adversarially trained model spreads reliance more evenly, reducing brittle feature dependencies. Standardized features make these ablations portable and interpretable across datasets.

Limitations & outlook. NF-UQ-NIDS-v2 is broad but still a curated family; adding raw-PCAP pipelines and more environments would strengthen external validity. Certification today focuses on specific tree ensembles; extending provable guarantees to flexible deep tabular models remains open—and valuable.

Bottom line. Robust, transferable NIDS require (1) cross-domain evaluation, (2) constraint-aware adversarial testing, and (3) a blend of empirical hardening and certified guarantees. NF-UQ-NIDS-v2 enables this discipline; our study shows it is both feasible and necessary.

6. Conclusion

This thesis presented a principled pipeline for evaluating and improving the adversarial robustness of ML-based network intrusion detection on structured (flow/tabular) data. Building on the NF-UQ-NIDS-v2 standard (43 NetFlow features with origin labels), we showed how to couple cross-domain evaluation with realizability-aware attacks so that reported robustness reflects operational constraints rather than unconstrained feature edits. Empirically, adversarially trained deep tabular models (e.g., TabTransformer) delivered the strongest *empirical* robustness at small/medium budgets, while verifiable tree ensembles provided *provable* guarantees (Certified Robust Accuracy and coverage) at practical certification cost—two complementary forms of assurance for high-stakes deployment. Finally, we highlighted the role of calibration (Brier/ECE) under attack and the importance of transparent threat-model policies, aligning with recent surveys that call for realistic, constraint-respecting NIDS evaluations.

Overall, the results recast the central question from "What is the IID score?" to "Is the detector robust, certifiable, and transferable across networks under feasible attacks?" The combination of standardized features, cross-domain testing, constraint-aware adversaries, and certification yields more defensible conclusions and clearer trade-offs for SOC decision-makers.

Recommendations. For future NIDS studies and deployments: (1) adopt NF-UQ-style standardized features and always report Source→Target results alongside IID; (2) publish a threat-model policy (immutability masks, ranges, validators) and use realizability-constrained adversaries; (3) pair adversarial training for empirical gains with certified trees for guarantees; (4) report calibration (ECE/Brier) and cost (latency/throughput) with robustness metrics; and (5) release code, configs, and seeds to enable end-to-end reproducibility.

References

[1] Bouzaachane, K., Guarmah, E. M. E., Alnajim, A. M., & Khan, S. (2025). Addressing Modern Cybersecurity Challenges: A Hybrid Machine Learning and Deep Learning Approach for Network Intrusion Detection. *Computers, Materials & Continua*, *84*(2).

[2] Calzavara, S., Cazzaro, L., Pibiri, G. E., & Prezza, N. (2023, November). Verifiable learning for robust tree ensembles. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security* (pp. 1850-1864).

[3] Cantone, M., Marrocco, C., & Bria, A. (2024). On the cross-dataset generalization of machine learning for network intrusion detection. *arXiv preprint arXiv:2402.10974*.

[4] Díaz-Bedoya, D., González-Rodríguez, M., Gonzales-Zurita, O., Serrano-Guerrero, X., & Clairand, J. M. (2025). Advanced Wind Speed Forecasting: A Hybrid Framework Integrating Ensemble Methods and Deep Neural Networks for Meteorological Data. *Smart Cities*, *8*(3), 94.
[5] Ennaji, S., De Gaspari, F., Hitaj, D., Kbidi, A., & Mancini, L. V. (2025). Adversarial challenges in network intrusion detection systems: Research insights and future prospects. *IEEE Access*.
[6] Ennaji, S., El Outa, A., Elaziz, M. A., & Cherkaoui, S. (2024). Adversarial challenges in network intrusion detection systems: A survey. *arXiv preprint arXiv:2409.18736.*
[7] Goldschmidt, P., & Chudá, D. (2025). Network intrusion datasets: a survey, limitations, and recommendations. *Computers & Security*, 104510.
[8] Gu, Z., Lopez, D. T., Alrahis, L., & Sinanoglu, O. (2024, April). Always be Pre-Training: Representation Learning for Network Intrusion Detection with GNNs. In *2024 25th International Symposium on Quality Electronic Design (ISQED)* (pp. 1-8). IEEE.
[9] Huang, X., Khetan, A., Cvitkovic, M., & Karnin, Z. (2020). Tabtransformer: Tabular data modeling using contextual embeddings. *arXiv preprint arXiv:2012.06678*.
[10] Layeghy, S., Sarhan, M., & Portmann, M. (2023). Explainable cross-domain evaluation of ML-based NIDS. *Computers & Electrical Engineering, 113*, 108841.
[11] Maseer, Z. K., Kadhim, Q. K., Al-Bander, B., Yusof, R., & Saif, A. (2024). Meta-analysis and systematic review for anomaly network intrusion detection systems: Detection methods, dataset, validation methodology, and challenges. *IET Networks*, *13*(5-6), 339-376.
[12] Park, C., & Lee, S. (2025). Tunable anisotropy in lattice structures via deep learning-based optimization. *International Journal of Mechanical Sciences*, *290*, 110121.
[13] Ruan, Y., Lan, X., Ma, J., Dong, Y., He, K., & Feng, M. (2024). Language modeling on tabular data: A survey of foundations, techniques and evolution. *arXiv preprint arXiv:2408.10548*.
[14] Sarhan, M., Layeghy, S., Moustafa, N., & Portmann, M. (2021). Towards a standard feature set of NIDS datasets. CoRR, abs/2101.11315. *arXiv preprint arXiv:2101.11315*.
[15] Sharma, S., & Chen, Z. (2024). A Systematic Study of Adversarial Attacks Against Network Intrusion Detection Systems. *Electronics*, *13*(24), 5030.
[16] TensorFlow. (2024). TensorFlow Lattice (TFL): Flexible, controlled, and interpretable lattice-based models. *Project documentation.* TensorFlow
[17] University of Queensland (UQ Cyber). (n.d.). Machine Learning-Based NIDS datasets (NF-UQ-NIDS-v2 portal). Retrieved 2023–2025. staff.itee.uq.edu.au
[18] Wang, X., Qiao, Y., Xiong, J., Zhao, Z., Zhang, N., Feng, M., & Jiang, C. (2024). Advanced network intrusion detection with tabtransformer. *Journal of Theory and Practice of Engineering Science*, *4*(03), 191-198.
[19] You, S., Ding, D., Canini, K., Pfeifer, J., & Gupta, M. (2017). Deep lattice networks and partial monotonic functions. *Advances in Neural Information Processing Systems (NeurIPS).*
[20] Zhao, J., Li, M., Zhao, X., & Yu, Z. (2024). Deep Isotonic Embedding Network: A flexible monotonic neural network. *Neural Networks, 172*, 106170.