International Journal of Safety and Security Engineering
Vol. 16, No. 10, October, 2026, pp. 2205-2215
Journal homepage: http://iieta.org/journals/ijsse

Unexamined Defaults Inflate IoT Intrusion Detection Results: A Protocol Audit on CIC IoT-DIAD 2024

Mohammad Alkhazaleh[1*], Mahmoud Baklizi[2], Salameh A. Mjlae[3,4], Musab Alzghoul[1], Loui Al Sardy[5]

[1] Department of Computer Science, Faculty of Information Technology, Isra University, Amman 11622, Jordan
[2] Department of Computer Science, Faculty of Information Technology, University of Petra, Amman 11196, Jordan
[3] Department of Cyber Security, Faculty of Information Technology, Jerash University, Jerash 26150, Jordan
[4] Department of Computer Science, Faculty of Information Technology, Al-Balqa Applied University, Al-Salt 19117, Jordan
[5] Department of Computer Science, Friedrich-Alexander-Universität Erlangen-Nürnberg, Erlangen 91058, Germany

Corresponding Author Email: m.alkhazaleh@iu.edu.jo

Copyright: ©2026 The authors. This article is published by IIETA and is licensed under the CC BY 4.0 license (http://creativecommons.org/licenses/by/4.0/).

https://doi.org/10.18280/ijsse.161002

Received: 24 August 2026
Revised: 22 September 2026
Accepted: 29 September 2026
Available online: 31 October 2026

Keywords:
class imbalance, evaluation methodology, experimental validity, intrusion detection, IoT security, random forest

ABSTRACT

Evaluation practice in flow-based Internet of Things (IoT) intrusion detection rests on defaults that are rarely examined: sampling a fixed number of records per capture file, splitting records at random, comparing class-balancing strategies without controlling training-set size, and accepting unbounded tree depth. This study audits all four on the CIC IoT-DIAD 2024 (19.5 million flows, 129 captures, eight families) and finds that each altered a conclusion drawn from these data, and in two cases reversed it. Per-file sampling makes a family's apparent share track its file count rather than its size, overrepresenting Mirai roughly 25-fold while compressing the genuinely dominant family to under a third of its weight. Under capture-aware evaluation across twenty draws, multiclass accuracy falls from 0.915 to 0.800 and macro-F1 from 0.617 to 0.491 over scorable families, while the ratio of standard deviations has a bootstrap 95% interval of 44.9–125.9, so a single-draw report cannot be interpreted without its draw composition. With training size held fixed and the comparison repeated over fifteen seeds, aggressive undersampling improves macro-F1 by 0.0625 (95% CI 0.0582–0.0662) over a size-matched random subsample, reversing the uncontrolled comparison. Removing the two features ranked highest by attribution costs 0.0027 macro-F1 while removing a lower-ranked pair costs 0.0250, so attribution rank determined necessity in neither direction. Capping tree depth raises benign recall from 0.323 to 0.977 under random splitting, whereas threshold calibration on a fixed model yields only 0.004 macro-F1. The corrected protocol and all code are released.

1. INTRODUCTION

In machine learning for network intrusion detection, reported accuracies on public Internet of Things (IoT) benchmarks now regularly exceed 0.95, so a new detector typically beats its predecessors by a fraction of a percentage point. Margins this small put the evaluation procedure itself under strain. Suppose some routine choice in the evaluation shifts results by more than the gap between two detectors: whichever one then comes out ahead, we have learned almost nothing about which is better.

None of this is a new worry. Fifteen years ago, Sommer and Paxson listed reasons why intrusion detection is hard to evaluate with machine learning; two of them were that a detector's output is not the same thing as an operational response, and that good benign traffic is scarce [1]. Arp et al. [2] went further and catalogued pitfalls that recur across security machine learning, with sampling bias, spurious correlations and badly chosen performance measures among them. Pendlebury et al. [3] demonstrated for malware classification that eliminating a single form of experimental bias—temporal leakage—substantially reduced reported performance.

What is less established is how large these effects are for flow-based IoT intrusion detection specifically, and which routine decisions matter most. This paper answers that question empirically for one widely used benchmark, CIC IoT-DIAD 2024, by auditing four decisions that are typically made without comment:

(1) How records are sampled from a corpus too large to use in full.

(2) How records are partitioned into training and test sets.

(3) How class balancing is compared, and whether training-set size is held constant when it is.

(4) Which model hyperparameters are left at library defaults, in particular tree depth.

Each of these is a default rather than a considered choice. Each materially altered a conclusion drawn from these data, and two reversed one.

We are not suggesting that earlier results were reported in bad faith, and our own work is not exempt. The distorted sampling we document was present in our own preliminary

analysis of this dataset, which we release alongside the corrected protocol. We use that analysis as the worked example precisely because we can report its internals in full and readers can reproduce them. What the paper offers is an estimate of what these defaults cost on this benchmark, together with a protocol that avoids them.

1.1 Contributions

Sampling. When a fixed number of records is drawn from each capture file, the share a family receives follows its number of files and not its number of flows. On CIC IoT-DIAD 2024, this overrepresents Mirai by approximately 25 times and compresses Denial of Service (DoS), the genuinely dominant family at 76.09% of flows, to under a third of its true weight.

Partitioning. Under a capture-aware protocol evaluated over twenty draws, multiclass accuracy falls from 0.915 ± 0.001 to 0.800 ± 0.112 and scored macro-F1 from 0.617 ± 0.008 to 0.491 ± 0.097. The ratio of standard deviations has a bootstrap 95% interval of 44.9–125.9, and the set of scorable families varies from three to six of eight between draws.

Benchmark limits. Two of eight families occupy a single capture file each and were not scorable in any of twenty draws under either capture-aware protocol. This is a property of the benchmark, not of any model trained on it.

Attribution. Attribution rank determined necessity in neither direction. Removing the pair ranked first and second by SHapley Additive exPlanations (SHAP) costs 0.0027 macro-F1 (95% CI 0.0014–0.0041), while removing a pair ranked sixth and twenty-seventh costs 0.0250 (0.0239–0.0261); removing an arbitrary pair costs 0.0006, an interval including zero.

Balancing. With training-set size held fixed and repeated over fifteen seeds, aggressive undersampling improves macro-F1 by 0.0625 (95% CI 0.0582–0.0662) over a same-size random subsample and falls short of a training set five times larger by only 0.0059.

Defaults. Tree depth affects minority-class recall far more than threshold calibration does, and scikit-learn's max_depth = None default yields the lowest benign recall of any setting tested under all three protocols, across twenty seeds. Holding the model fixed, threshold calibration yields 0.004 macro-F1 and merely redistributes error within the minority class. We also describe a threshold-selection rule that looks sensible but becomes degenerate when imbalance is extreme.

Zero-shot observation. In the draws where an attack family reached the test set without any training examples, most of its flows were assigned to some other attack family and comparatively few to benign, so binary detection held up better than family attribution. Sections 5.1–5.7 take these findings one at a time.

1.2 Scope

The audit covers one dataset and one classifier family. We expect the mechanisms to carry over, since none of them relies on anything peculiar to random forests or to this corpus. The magnitudes are another matter: they belong to CIC IoT-DIAD 2024 and should be treated as a single observed case. We are not proposing a new detection method, and we do not claim that the performance figures below are the best that can be achieved on this dataset.

2. BACKGROUND AND RELATED WORK

2.1 Benchmark construction and previous studies

CIC IoT-DIAD 2024 was released by the Canadian Institute for Cybersecurity as a dual-function corpus supporting both device identification and anomaly detection [4]. It follows earlier IoT traffic benchmarks such as Bot-IoT [5] and CICIoT2023 [6] in providing labelled attack captures collected from a physical testbed. It is distributed in two feature representations: a flow-based directory extracted with CICFlowMeter, and a packet-based directory derived from per-packet analysis. All experiments in this paper use the flow-based representation, which comprises 129 capture-derived CSV files and 19,519,162 flows. We state this explicitly because studies using the packet-based representation report substantially different corpus dimensions—one such study reports 35,672,407 records across 180 files [7]—and the two are not directly comparable.

The corpus is organised into eight class folders, one benign and seven attack families, among them Mirai, whose original outbreak and measurement are documented in the study [8]. Ground-truth labels are therefore folder-derived: the dataset's own Label column carries the placeholder value NeedManualLabel throughout. This has a consequence rarely acknowledged in work using the dataset, namely that benign traffic captured during an attack window is labelled as attack. Comparable labelling and flow-construction defects have been documented for CICIDS2017 [9] in a detailed re-examination of that corpus [10], indicating that such issues recur across this family of corpora rather than being isolated.

The corpus was released together with a lightweight integrated framework for simultaneous device identification and anomaly detection [4], and subsequent work on it has been predominantly architectural: hybrid convolutional neural network (CNN)-XGBoost models with SHAP-based interpretability [11], one-dimensional CNN, long short-term memory (LSTM), recurrent neural network (RNN) and multilayer perceptron (MLP) comparisons reporting 99.12% multi-class accuracy for the 1D CNN [12], and autoencoder-based anomaly detection with transfer learning for edge deployment [13]. Reported accuracies cluster in the 0.93–0.99 range. Such clustering is itself informative: when differences between methods fall below a percentage point, the evaluation protocol rather than the architecture becomes the factor determining the ranking.

Two recent studies engage directly with evaluation validity on this corpus, and this work is positioned against both.

The first concerns feature selection: ranking predictors on the full dataset rather than on training-fold data alone materially changes reported multi-class performance, and a leakage-aware protocol on this corpus yields 0.9324 accuracy with 0.5910 macro-F1 under matched three-fold cross-validation [14]. Those figures are the closest published comparison to the random-split results reported in Section 5.2 (0.9154 accuracy, 0.6174 macro-F1), obtained with a different model family, which suggests the baseline used here is representative rather than idiosyncratic. That protocol nonetheless partitions at the record level, so the generalisation gap quantified here is outside its scope.

The second, and the closest prior work, is a preprocessing-governance framework for this corpus [7] combining device-disjoint split construction, quantified leakage auditing via flow-signature Jaccard analysis, train-only transformation,

shared-safe feature selection, and explicit acceptance gates, reporting integrity metrics including a Device-Feature Dependency Index of zero and stronger preprocessing integrity than random stratified splitting. Its central premise—that reported detector behaviour cannot be interpreted rigorously if the data substrate is not governed rigorously—is one this study shares.

The present study differs in object, unit, and claim. That framework deliberately says nothing about downstream classifier performance; it reports integrity metrics for an auditable preprocessing substrate. Our study measures something complementary, namely how far reported detection performance moves when the protocol changes, and how variable it becomes. The grouping unit also differs: device-level grouping enriched with vendor metadata, against the capture file used here. The distinction matters because capture files in this corpus differ in size by three orders of magnitude, which is the mechanism behind the variance we report and which device-level grouping does not surface. Finally, sampling is not examined there, whereas the distortion documented in Section 5.1 arises before any split is constructed and is therefore orthogonal to split governance. The two studies address successive stages of the same problem.

2.2 Partitioning and experimental validity

Structural obstacles to applying machine learning in intrusion detection were identified early, among them the difficulty of obtaining representative benign traffic and the mismatch between classification metrics and operational value [1]; methodological guidelines for malware experiments followed, emphasising transparency about dataset composition [15], and a systematisation of ten recurring pitfalls across security machine learning now covers sampling bias, spurious correlation and base-rate neglect [2]. The closest precedent to our work is the finding that splitting malware samples at random lets future information leak into training, and that reported performance falls substantially once this temporal bias is removed [3].

That precedent concerns temporal independence in malware classification; the concern here is capture independence in network flows. The two differ in one respect that turns out to matter: time provides a natural total ordering and an approximately uniform sampling density, whereas capture sessions are discrete units of wildly unequal size. Group-aware splitting is therefore not merely stricter but also substantially higher-variance, a property that does not appear to have been quantified for IoT flow benchmarks. The principle that dependent observations require group-aware rather than random partitioning is long established outside security, notably in ecological modelling where blocked cross-validation is standard for spatially or hierarchically structured data [16], and a general formulation of leakage and the conditions under which it arises is available [17].

2.3 Class imbalance, metric choice, and feature attribution

Resampling remains the dominant response to class imbalance in intrusion detection, with the synthetic minority over-sampling technique (SMOTE) [18] and its variants widely applied, alongside undersampling and cost-sensitive learning; the field has been surveyed comprehensively [19]. The choice of evaluation measure is equally consequential under skew: receiver operating characteristic (ROC) analysis is over-optimistic relative to precision-recall when the negative class dominates [20], and the Matthews correlation coefficient has been argued preferable to accuracy and F1 in imbalanced binary settings [21]. Reported comparisons frequently find that aggressive undersampling degrades performance.

Such comparisons typically alter class balance and training-set size simultaneously, since undersampling necessarily discards records. Where no size-matched control is reported, an observed degradation cannot be attributed to balancing rather than to data loss. Section 5.5 shows that on this dataset the attribution reverses once the control is added.

SHAP [22] provides locally accurate additive feature attributions with desirable theoretical properties and is now routinely used in intrusion detection work, including on this dataset [11], to argue that a model has learned meaningful behaviour rather than dataset artefacts.

This inference does not follow. Attribution measures a feature's contribution to the model as fitted, which for a correlated feature set reflects which of several interchangeable features the learner happened to select—a known property of impurity-based and Shapley-based rankings on tree ensembles [23]. Systematic bias in random-forest importance measures has been demonstrated and traced to the splitting criterion itself [24], and the extension of SHAP to tree ensembles was accompanied by the observation that attribution and necessity are distinct quantities [25]. Section 5.4 provides a direct demonstration on this corpus, in both directions: the pair of features ranked highest by attribution can be removed at negligible cost, while a lower-ranked pair cannot, and the ranking itself shifts under a change of a single hyperparameter. Establishing necessity requires an ablation with controls, since removing any two of a set of correlated features carries some cost and only a control reveals how much.

3. THREATS TO VALIDITY

An audit of evaluation practice invites the question of whether its own protocol is sound. We state the limitations of this study before presenting results, and return to those that bear on interpretation in Section 6.

3.1 Construct and internal validity

Sampling floor. Our corrected sample is proportional to true family size with a floor of 2,000 flows per family. Without a floor, Brute Force (0.02% of the corpus) and Web-Based (0.06%) would contribute fewer than 100 flows each and could not be modelled at all. The floor therefore represents four families—Brute Force, Web-Based, Spoofing and Mirai—above their true proportions, and our sample is not a faithful miniature of the corpus. It is, however, a documented departure with a stated rule, which is the property we argue per-file sampling lacks. Results for the four-floored families may therefore be optimistic relative to strictly prevalence-proportional sampling, and should be interpreted with that in mind.

Label provenance. Ground-truth labels derive from the folder containing each capture; the dataset's own Label column holds the placeholder value NeedManualLabel for every record. A benign flow recorded during an attack capture therefore ends up labelled as attack. This comes with the benchmark; we did not choose it. We suspect it is part of the

reason benign-class recall is low, yet the information available does not let us tell label noise apart from model error. Every study on this dataset has the same problem, though it is seldom stated.

Metric choice. At a 1:50 class ratio, accuracy mostly reflects the majority class, so macro-F1 is reported alongside it throughout. A family with zero test support in some draw gets an undefined recall for that draw; we do not count it as zero, and we exclude it from macro averages. Several widely used implementations silently put zero in place of undefined, and the resulting figures look measured when in fact they are not.

Single classifier family. We used random forests for everything. The mechanisms we identify are not specific to tree ensembles: per-file sampling distorts class priors regardless of learner, and capture-aware splitting removes shared session context regardless of learner. The depth finding is specific to tree-based models, though the general point—that an unexamined library default can dominate a carefully tuned intervention—is not.

Single dataset. We audit CIC IoT-DIAD 2024. The magnitudes we report are specific to its file organisation, and a corpus with uniform file sizes and several captures per family would show smaller effects. We claim the mechanism generally and the magnitudes locally.

Uncalibrated probabilities. Thresholds in Section 5.6 are applied to raw random-forest scores, which are known to be poorly calibrated [26]. Probability calibration might change the shape of the threshold curve, though it would not affect the depth comparison, where all models are evaluated at the same decision rule.

3.2 External and conclusion validity

Some of what we would like to measure cannot be measured on this corpus at all. Recon and Brute Force were each recorded in a single capture file, Benign in four. A group-aware split keeps every capture whole, so in most draws each of these families lands entirely in training or entirely in test. In our twenty draws Recon and Brute Force were never scorable; Benign was scorable 11 times. Running more draws would not help, because the obstacle is the file structure of the benchmark. Per-family conclusions drawn from CIC IoT-DIAD 2024, including those in Section 5.3, should be read with that in mind.

Number of draws. Twenty capture-aware draws were run for the protocol comparison and the per-family analysis, and fifteen for the balancing and ablation experiments. Between-draw variation under capture-aware splitting is large (eight-class accuracy SD 0.1117, against 0.0014 for random splitting), so intervals remain wide at twenty draws, and ratios and differences are reported with bootstrap confidence intervals. Benign traffic was present in the test partition in 11 of the twenty draws. Benign figures in Sections 5.2 and 5.3 are computed over these 11 draws, and the count is restated where they appear. In the depth sweep, 11 draws were usable under file-proportion splitting and 9 under size-matched splitting.

Formal significance tests [27] require results over multiple datasets and were not applied to this single-corpus audit. For the repeated stochastic comparisons behind our inferential claims we use bootstrap intervals, and we do not interpret a difference whose interval contains zero. One consequence is that we make no claim about a tree depth that would be optimal regardless of protocol. Under capture-aware evaluation the best settings have heavily overlapping intervals. Random splitting does single out depth 30 within the tested grid, but we do not assume that this optimum transfers to other protocols. Every reported difference is accompanied by its variance, and where the variance is too large to support a claim we say so and go no further.

4. EXPERIMENTAL SETUP

4.1 Corpus and sampling

We use CIC IoT-DIAD 2024, which comprises 129 capture-derived flow files organised into eight class folders. A full enumeration of the corpus yields 19,519,162 flows. Table 1 reports the per-family counts.

Table 1. Family composition of CIC IoT-DIAD 2024

Family	Files	File Share	True Flows	True Share
DoS	27	20.93%	14,853,087	76.09%
DDoS	61	47.29%	3,478,814	17.82%
Recon	1	0.78%	442,158	2.27%
Benign	4	3.10%	398,330	2.04%
Mirai	29	22.48%	174,588	0.89%
Spoofing	3	2.33%	157,238	0.81%
Web-Based	3	2.33%	11,328	0.06%
Brute Force	1	0.78%	3,619	0.02%
Total	129	100%	19,519,162	100%

Note: Under a uniform per-file sampling quota, the sampled distribution converges to the file-share column rather than the true-share column. DoS: Denial of Service; DDoS: Distributed Denial of Service; Recon: reconnaissance.

Two features of the corpus shape the rest of the analysis. The first is that the number of files a family has says little about how prevalent it is. DoS accounts for 76.09% of all flows in 27 files, Distributed Denial of Service (DDoS) for 17.82% in 61 files, and Mirai for just 0.89% in 29 files. The second is that some families rest on very few captures (one each for Recon and Brute Force, four for Benign), so a partition by capture cannot treat all families alike.

Labels are taken from the folder in which each capture is stored, since the Label column shipped with the dataset holds the placeholder NeedManualLabel for every record. As a result, benign background traffic recorded during an attack capture carries an attack label.

A common practice in earlier work is to take a fixed number of records from every capture file, which ties a family's sampled share to how many files it has. We sample in proportion to true family size instead, with at least 2,000 flows per family so that the small families can still be modelled. This gives a working sample of 125,861 flows. Four families (Brute Force, Spoofing, Web-Based, Mirai) are at or close to the floor and so appear above their real share of the corpus; we state this openly and do not describe the sample as purely proportional.

Each record we sampled carries the filename of its source capture, and capture-aware partitioning relies on this. Appendix A lists the sampling rule, the software environment and every experimental configuration. One side effect of proportional allocation is rounding: each file's quota has to be a whole number, and for the smallest files in the large families it rounds to zero. In the end 124 of the 129 captures contribute to the working sample (22 of the 27 DoS files, for instance), so every capture-aware result is computed over 124 groups.

4.2 Preprocessing, model and evaluation protocols

Identifier and temporal columns (Flow ID, Src IP, Dst IP, Timestamp) are dropped, as is the placeholder Label column. We keep the numeric features, convert infinities to missing values and fill missing values with the column median. Seven columns take a single value across the whole sample and are removed: Bwd PSH Flags, Fwd URG Flags, Bwd URG Flags, URG Flag Count, Fwd Bytes/Bulk Avg, Fwd Packet/Bulk Avg, and Fwd Bulk Rate Avg. That leaves 72 features. Exact duplicate flows are also removed, but there are only three of them (0.00%), so leakage through duplicates across partitions is not an issue here.

In all experiments the classifier is a random forest with 300 trees (200 in the depth sweep) and class_weight = "balanced_subsample", using the scikit-learn implementation [28]. Except where depth is the variable under study, we set max_depth = 25; Section 5.6 explains why the library default is inappropriate here and why we report a single depth across all tables.

We compare three partitioning protocols: one conventional random-record split and two capture-aware variants. Under the random-record protocol, records are shuffled and divided 80/20 with stratification by class. Under file-proportion capture-aware splitting, whole capture files are held out using the capture filename as the grouping variable, targeting approximately 20% of files. Under size-matched capture-aware splitting, whole captures are accumulated until approximately 20% of records are assigned to the test partition. Both capture-aware variants preserve capture independence, so no test flow shares a capture with any training flow; they differ only in what quantity the 20% target applies to. A single capture-aware draw is not stable, so the protocol comparison and the per-family evaluation are run over twenty independent draws for each capture-aware protocol. We report means with standard deviations and, for each claimed ratio or difference, a bootstrap 95% confidence interval based on 10,000 resamples. The balancing and feature-removal experiments use fifteen seeds and paired per-seed differences, and the depth sweep uses twenty draws under all three protocols.

5. RESULTS

5.1 Per-file sampling makes class balance a function of file count

If every file contributes the same number of records, each family's sampled share approaches its share of capture files, regardless of how large the family is. Table 1 places the two side by side.

The resulting distortion is large. Mirai holds 22.48% of capture files but 0.89% of flows, an overrepresentation of roughly 25-fold. DoS holds 20.93% of files but 76.09% of flows and is compressed to under a third of its true weight. DDoS, at 47.29% of files and 17.82% of flows, is inflated by a factor of 2.7.

We can demonstrate the effect concretely because we applied this protocol ourselves before auditing it. Our preliminary run, whose code and build summary are included in the released repository, placed DDoS at 49.19%, Mirai at 23.39% and Benign at 3.23%, closely tracking the file-share distribution (47.29%, 22.48%, 3.10%) and bearing no relation to the true one. For seven of the eight families, the sampled share was within two percentage points of the file share. DoS was off by 3.19 points, for the reason already behind the 124-of-129 figure in Section 4.2: five DoS captures contain fewer records than the per-file quota, so DoS received 22,000 records out of a possible 27,000 and the other families' shares rose slightly to compensate. The class distribution we would have reported was measuring the folder layout.

The mechanism is simple. Benign packs about 100,000 flows into each of its four files, whereas Mirai's 174,588 flows are spread over twenty-nine. A quota of n records per file favours families split into many files and penalises those concentrated in a few. Figure 1 places the two distributions side by side.

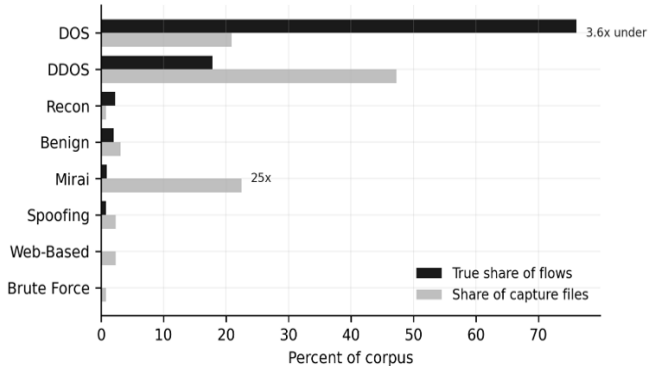

Figure 1. Share of capture files against true share of flows, by family

Note: Under per-file sampling the grey bars determine the sampled distribution. DoS: Denial of Service; DDoS: Distributed Denial of Service; Recon: reconnaissance.

Under per-file sampling, conclusions about class imbalance, minority-class behaviour or confusion between classes largely reflect the way the files are organised. One such conclusion reverses entirely once the sample is corrected.

5.2 Capture-aware evaluation: Lower performance, far higher variance

Table 2 compares three partitioning protocols across twenty independent draws. The two capture-aware protocols differ only in how held-out captures are chosen: the first holds out 20% of capture files, the second accumulates captures until approximately 20% of records are held out.

Multiclass accuracy falls by 11.5 points under file-proportion splitting and 7.7 points under size-matched splitting. The macro-F1 gap depends on how unscorable families are handled: over all eight families it falls from 0.617 to 0.292, but 0.199 of that drop comes from families that had zero test support or zero training exposure in a given draw and were scored as 0.0000 by the zero-division convention. Over families actually present in both partitions, the fall is from 0.617 to 0.491, a relative reduction of 20% rather than 53%. We give both figures, since the gap between them shows how much the single-capture families matter, and that is a result in its own right.

The variance turns out to be the more important result. Over twenty draws, the ratio of standard deviations between capture-aware and random splitting has a bootstrap 95% interval of 44.9–125.9 for multiclass accuracy under file-proportion splitting and 35.9–93.5 under size-matched splitting; for the binary task the intervals are 7.5–30.7 and 7.6–20.0. These intervals are wide, and we report them rather than

the sample ratios they enclose. Across all four comparisons, the lower bound of the bootstrap 95% interval exceeds seven, indicating substantially greater variability under capture-aware evaluation than under random splitting of the same data. Fixing the test size reduces the multiclass accuracy standard deviation from 0.1117 to 0.0842, a reduction of about a quarter. Variability in test-set size therefore accounts for part of the excess variance but not most of it; the remainder is associated with which captures fall on each side of the split, though the two cannot be separated completely because the size-matched protocol also changes which captures are selected.

Table 2. Performance under three partitioning protocols, mean ± standard deviation over twenty draws, at max_depth = 25

Task	Protocol	Accuracy	Macro-F1 (All 8)	Macro-F1 (Scored)
Binary	Random record	0.9626 ± 0.0018	0.7009 ± 0.0075	—
Binary	Capture-aware (file-proportion)	0.9550 ± 0.0309	0.7020 ± 0.0298	(11 of 20 draws)
Binary	Capture-aware (size-matched)	0.9653 ± 0.0218	0.6436 ± 0.0587	(9 of 20 draws)
8-class	Random record	0.9154 ± 0.0014	0.6174 ± 0.0080	0.6174 ± 0.0080
8-class	Capture-aware (file-proportion)	0.8001 ± 0.1117	0.2924 ± 0.0376	0.4914 ± 0.0968
8-class	Capture-aware (size-matched)	0.8387 ± 0.0842	0.2539 ± 0.0567	0.4977 ± 0.0852

Note: Macro-F1 (Scored) averages only over classes present in both partitions of a given draw, with the count of such draws given in that column; under random splitting every class is scored in every draw, so the two macro-F1 columns coincide. F1: harmonic mean of precision and recall; max_depth: maximum tree depth; ±: one standard deviation about the mean.

The mechanism is mechanical. GroupShuffleSplit partitions files, not records, and file sizes span three orders of magnitude, so a fixed proportion of files does not yield a fixed proportion of data. Over the twenty draws, a 20% file hold-out gave test partitions containing anywhere from 8.0% to 40.3% of the records, and the largest test class switched between DoS and DDoS. The size-matched protocol keeps this proportion between 20.0% and 28.5%, which cuts the range from 32.3 points to 8.5 while still treating the capture as the unit of independence.

The draws show the same instability in their class composition. Benign traffic was present in the test partition in 11 of 20 draws under file-proportion splitting and 9 of 20 under size-matched splitting, because Benign occupies only four captures; Recon and Brute Force, each resting on a single capture, were never scorable. Figure 2 shows the resulting per-seed spread.

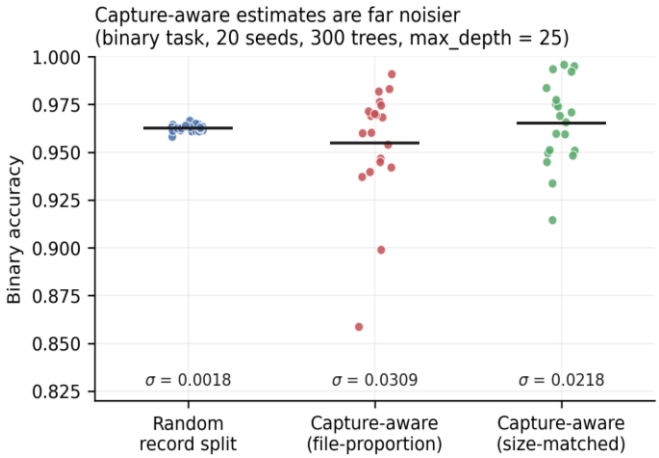

Figure 2. Per-seed binary accuracy under the three protocols, from the same twenty-seed run as Table 2 (300 trees, max_depth = 25)

Note: Horizontal bars mark the mean. max_depth: maximum tree depth.

Implication. A capture-aware result on this corpus should be reported as a distribution together with the composition of the draws that produced it, since test size varied from 8.0% to 40.3% of records across twenty draws. Conversely, a random-split result reports a stability the underlying data does not possess.

5.3 Per-family behaviour under capture-aware evaluation

Table 3 reports per-family recall across twenty capture-aware draws under file-proportion splitting, averaged only over draws in which the family was present in both partitions. Recon and Brute Force each occupy a single capture file, so every draw places the entire family on one side of the split: neither was scorable in any of the twenty draws under either protocol. Benign occupies four captures and was scorable in 11 draws; Spoofing in 14 and Web-Based in 13. These are not measurements of 0.0000—they are undefined, and we report them as such rather than allowing a zero-division convention to enter an aggregate.

Table 3. Per-family recall under capture-aware (file-proportion) evaluation over twenty draws, max_depth = 25, averaged over scorable draws only

Family	Mean Recall	σ	Draws Scored	Mean Support
DoS	0.9294	0.0757	20 of 20	19,072
DDoS	0.6946	0.2326	20 of 20	5,554
Mirai	0.2862	0.0628	20 of 20	348
Spoofing	0.2170	0.1057	14 of 20	597
Benign	0.2872	0.1144	11 of 20	1,090
Web-Based	0.2654	0.0992	13 of 20	550
Recon	n/a	—	0 of 20	—
Brute Force	n/a	—	0 of 20	—

Note: Support is averaged over scored draws. DoS: Denial of Service; DDoS: Distributed Denial of Service; Recon: reconnaissance.

The protocol changes what is measured, not only what is measured as. The number of scorable families varied from three to six out of eight across the twenty draws; in some draws only DoS, DDoS and Mirai could be evaluated at all. Two studies that both adopt capture-aware evaluation on this corpus may therefore be reporting macro-averages over different family sets, and their macro-F1 values are not comparable unless the scored set is stated. Random splitting hides this problem, because every family is scorable in every draw.

Among families that could be scored, four of six fall below 0.30 recall and only DoS exceeds 0.75. Benign was scorable in 11 of 20 draws, reaching 0.2872 (95% CI 0.2250–0.3538); under random splitting on the same data, it reaches 0.6972. At a 1:50 class ratio, a classifier that labels everything “attack”

already scores 0.98 accuracy, and standard accuracy reporting cannot tell this apart from real detection.

The limitation lies in the dataset and would apply to any model. Two of the eight families in CIC IoT-DIAD 2024 have only one capture each, so they cannot be evaluated with capture-aware splitting, whatever the sample size, seed, or number of repetitions. Only additional captures would fix this.

5.4 Attribution rank does not determine necessity, in either direction

In an unconstrained forest, Src Port is third by mean absolute SHAP value, and it is the only identifier-derived feature in the top ten. This alone does not show that the source port encodes attack behaviour. Engelen et al. [10] found that on CICIDS2017 destination port alone gave high accuracy, since port values identify a flow's capture of origin rather than any malicious property. Whether the model depends on them is an empirical question that attribution rank does not answer, so we removed them. A drop is expected when any two of 72 correlated features are removed, so two control pairs were removed in the same way: Flow IAT Min with Fwd IAT Min (SHAP ranks 1 and 2) and Flow IAT Std with Fwd Packet Length Mean (an arbitrary pair). Each removal was repeated over fifteen seeds; Table 4 gives paired per-seed differences with bootstrap 95% intervals.

Table 4. Feature-pair ablation (fifteen seeds)

Task	Pair Removed	Depth	Macro-F1 Drop	95% CI	Benign Recall Drop
8-class	Src Port, Dst Port	25	0.0250	0.0239–0.0261	—
8-class	Flow IAT Min, Fwd IAT Min	25	0.0027	0.0014–0.0041	—
8-class	Flow IAT Std, Fwd Pkt Len Mean	25	0.0006	-0.0006–0.0016	—
8-class	Src Port, Dst Port	None	0.0237	0.0224–0.0251	—
8-class	Flow IAT Min, Fwd IAT Min	None	0.0007	-0.0002–0.0016	—
8-class	Flow IAT Std, Fwd Pkt Len Mean	None	0.0001	-0.0008–0.0010	—
Binary	Src Port, Dst Port	25	-0.0037	-0.0063 to -0.0011	0.0347
Binary	Flow IAT Min, Fwd IAT Min	25	-0.0002	-0.0019–0.0015	-0.0003
Binary	Flow IAT Std, Fwd Pkt Len Mean	25	0.0015	0.0005–0.0026	-0.0016

Note: Paired mean differences, fifteen seeds, bootstrap 95% intervals; positive values denote a drop after removal. Src: source; Dst: destination; IAT: inter-arrival time; Fwd: forward; Pkt: packet; Len: length; Std: standard deviation; CI: confidence interval; F1: harmonic mean of precision and recall; max_depth: maximum tree depth, None denoting unbounded depth.

Removing the port pair reduces eight-class macro-F1 by 0.0250 (95% CI 0.0239–0.0261) at depth 25 and by 0.0237 (0.0224–0.0251) at unbounded depth. Removing the top-ranked timing pair costs 0.0027 (0.0014–0.0041), and removing an arbitrary pair costs 0.0006 (-0.0006–0.0016), an interval that includes zero. The port-pair interval is clearly separated from both control intervals.

The port features, then, are not redundant. Of the pairs tested, they are the only ones whose removal costs clearly more than dropping two features at random, and the cost is about nine times that of removing the two highest-ranked features. The binary task shows how easily an aggregate score hides this. At depth 25 the binary macro-F1 even improves slightly without the ports (0.7012 to 0.7050). Benign recall, however, decreases from 0.7018 to 0.6671 (paired drop 0.0347, 0.0303–0.0396), compared with changes of at most 0.002 for either control pair. The ports thus contribute minority-class recall at a small cost to the majority class, which the binary macro-F1 does not reveal.

The ranking is also sensitive to depth. With max_depth = 25 instead of unbounded (same tree count, split and SHAP sample), Src Port moves from rank 3 to rank 6 under SHAP and from rank 9 to rank 13 under impurity importance; Flow IAT Min is first in both settings. Spearman correlation between the SHAP and impurity rankings is 0.8823 (unbounded) and 0.8975 (depth 25). Both rankings are computed from the same forest, so this agreement is not independent evidence.

Implication. Attribution rank and necessity disagreed in both directions here: the features ranked first and second could be dropped for 0.0027 macro-F1, while dropping those ranked sixth and twenty-seventh cost at least 0.0250. A paper that wants to argue its model learned meaningful behaviour needs a removal test with controls; otherwise a feature the model really depends on looks no different from one that merely ranks high among correlated alternatives.

5.5 With training size held fixed, balancing helps

Aggressive undersampling is often reported to hurt classification performance. The trouble with that comparison is that undersampling also throws data away, so the balanced model is trained on far fewer records than the one it is compared with. Our third arm pulls the two apart. It is a plain random subsample of the training set, with the original class proportions, cut to 20,126 records so that it matches the balanced arm exactly. Over fifteen seeds we redrew both the split and the subsample each time, so the intervals include both sources of variation. Table 5 has the results for all three.

Table 5. Undersampling versus a size-matched random subsample, 8-class task, max_depth = 25

Arm	Train n	Macro-F1 (mean ± SD)	95% CI
A—full training set, original balance	100,686	0.6178 ± 0.0087	0.6132–0.6218
B—undersampled to 3× smallest family	20,126	0.6120 ± 0.0062	0.6087–0.6147
C—random subsample, size-matched to B	20,126	0.5495 ± 0.0052	0.5471–0.5521

Note: Fifteen seeds, 300 trees, random record-level splitting. F1: harmonic mean of precision and recall; n: number of training records; SD: standard deviation; CI: confidence interval; max_depth: maximum tree depth.

With the same amount of training data, the balanced arm beats the size-matched control by 0.0625 macro-F1 (95% CI 0.0582–0.0662), and the margin is about the same in each of the fifteen seeds. Against the full arm, trained on five times as many records, the balanced arm is down just 0.0059 (0.0035–0.0081). We take that gap seriously, but it is about a tenth of what balancing gained. Most of the drop seen when the full arm is set against an uncontrolled subsample is therefore the price of training on less data.

Change balance and size together and the size effect ends up booked to balance; holding size fixed is what prevents that here.

5.6 Benign recall: tree depth versus threshold calibration

We started with the threshold, since that is where most papers go when minority recall collapses. Maximising macro-F1 on a held-out validation split gave $\tau = 0.38$ and moved macro-F1 from 0.6929 to 0.6972, a gain of 0.004. Had we tuned on the test set instead, we would have gained 0.0003 ($\tau = 0.34$, macro-F1 0.6975), so selecting on validation data costs essentially nothing. Table 6 shows the three operating points.

The gain is small, and it is not the kind of gain one wants. Benign precision went up 3.3 points (0.2926 to 0.3254) while benign recall went down 10.4 (0.6592 to 0.5551): minority errors were traded, not reduced. At 1:50 there is little else a threshold can do. Our own first attempt makes the point more sharply. We had chosen τ with the rule $0.7 \cdot F1_attack + 0.3 \cdot benign_recall$, which climbed all the way to the end of the sweep and settled on $\tau = 0.98$; there benign precision is 0.164, macro-F1 is 0.614, and every balanced metric is worse than at the untouched default. The cause is the imbalance itself. Attack F1 stays above 0.94 wherever τ sits, so a rule that pays for minority recall and never charges for minority precision will always run to the extreme. Macro-F1 does charge for it.

Depth had a much larger effect than the threshold did. The two effects were measured under different protocols, so we state them separately. In the depth sweep of Table 7 (same tree count, seeds and split), relaxing the depth limit from 25 to the unbounded default lowers binary benign recall from 0.6949 (0.6859–0.7041) to 0.3230 (0.3155–0.3312) under random splitting, 37.2 points, and from 0.4222 to 0.1635 under capture-aware splitting. Within the fixed depth-25 model of Table 6, moving the threshold from 0.50 to the validation-selected 0.38 changes benign recall by −10.4 points and macro-F1 by 0.004. Regularisation, not the decision threshold, is what recovers minority-class performance here. Table 7 reports the max_depth sweep under the random-record and file-proportion capture-aware protocols over twenty seeds. Under the size-matched protocol benign recall falls in the same order; those numbers are in the repository.

Table 6. Operating points for threshold calibration, binary task

Operating Point	Accuracy	Benign Prec.	Benign Recall	Attack Recall	Macro-F1
Default $\tau = 0.50$	0.9623	0.2926	0.6592	0.9684	0.6929
Calibrated $\tau = 0.38$	0.9689	0.3254	0.5551	0.9771	0.6972
Degenerate criterion $\tau = 0.98$	0.9026	0.1640	0.9776	0.9011	0.6143

Note: Row three is the τ picked by the composite rule described in the text. Prec.: precision; F1: harmonic mean of precision and recall; τ: threshold on the predicted attack probability.

Table 7. Benign recall and macro-F1 by max_depth, binary task (twenty seeds, bootstrap 95% intervals)

Max_Depth	Random: Benign Recall	Random: Macro-F1	Capture (file-prop.): Benign Recall	Capture (file-prop.): Macro-F1
6	0.9774 (0.9745–0.9804)	0.5830 (0.5817–0.5845)	0.9125 (0.8959–0.9305)	0.6474 (0.5976–0.7033)
10	0.9413 (0.9377–0.9449)	0.6261 (0.6249–0.6273)	0.7800 (0.7267–0.8362)	0.6794 (0.6388–0.7242)
12	0.9094 (0.9049–0.9135)	0.6440 (0.6426–0.6453)	0.6896 (0.6089–0.7713)	0.6914 (0.6591–0.7287)
20	0.7856 (0.7786–0.7927)	0.6821 (0.6797–0.6846)	0.5141 (0.4433–0.5890)	0.7068 (0.6860–0.7268)
25	0.6949 (0.6859–0.7041)	0.7004 (0.6972–0.7035)	0.4222 (0.3542–0.4942)	0.7018 (0.6841–0.7186)
30	0.6030 (0.5946–0.6118)	0.7216 (0.7178–0.7253)	0.3250 (0.2643–0.3890)	0.6804 (0.6601–0.7022)
None	0.3230 (0.3155–0.3312)	0.6970 (0.6931–0.7012)	0.1635 (0.1211–0.2081)	0.6102 (0.5809–0.6400)

Note: 200 trees per forest. Random-split columns average all twenty draws. Capture-aware columns average only the 11 draws whose test partition held benign traffic. Bootstrap 95% intervals in parentheses. file-prop.: file-proportion capture-aware protocol; max_depth: maximum tree depth (None = unbounded scikit-learn default); F1: harmonic mean of precision and recall; CI: confidence interval.

We plot the two curves in Figure 3. Under random splitting, benign recall falls at every step, from 0.9774 (0.9745–0.9804) at depth 6 to 0.3230 (0.3155–0.3312) at the library default; trees left unbounded grew to a mean depth of 60.6. Macro-F1 peaks at depth 30 (0.7216, 0.7178–0.7253) and is lower at the default (0.6970, 0.6931–0.7012). The same monotone ordering holds under capture-aware splitting, where benign recall decreases from 0.9125 (0.8959–0.9305) at depth 6 to 0.1635 (0.1211–0.2081) at the default across the 11 draws in which benign traffic was testable.

Two conclusions are supported and one is not.

Supported: the unbounded default minimises minority recall under every protocol tested. Benign recall at max_depth = None is 0.3230 (0.3155–0.3312) under random splitting, 0.1635 (0.1211–0.2081) under file-proportion capture-aware splitting and 0.2107 (0.1654–0.2498) under size-matched splitting. It is the lowest value in the grid under all three, and its interval does not approach that of any other setting. The unbounded setting is not a single fixed configuration: mean realised depth was 60.6 under random splitting, 58.7 under file-proportion capture-aware splitting and 58.6 under size-matched splitting, so it varies with the training partition. For macro-F1 the pattern is protocol-dependent.

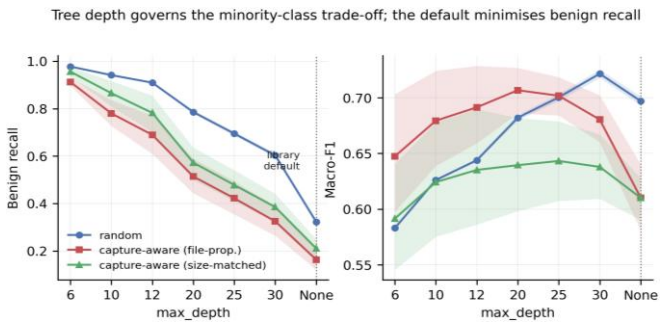

Figure 3. Benign recall and macro-F1 against maximum tree depth under the three protocols

Note: The rightmost tick represents the unbounded default depth. Shaded bands indicate bootstrap 95% confidence intervals. F1: harmonic mean of precision and recall; max_depth: maximum tree depth; CI: confidence interval.

Under file-proportion capture-aware evaluation, the unbounded default yields the lowest macro-F1 (0.6102), against 0.6474–0.7068 across the bounded depths tested. Under size-matched capture-aware evaluation, however, depth 6 yields the lowest (0.5915), while the default reaches 0.6102; and under random splitting the default is mid-range (0.6970) while depth 6 performs substantially worse (0.5830). The claim that is consistent across all three protocols therefore concerns minority-class recall rather than macro-F1. max_depth = None is scikit-learn's default and is rarely reported. Probst et al. [29] observe that random forests generally perform reasonably at library defaults, which is part of why those defaults go unexamined; our results indicate that under severe imbalance the default is nonetheless the least favourable setting for the minority class.

Supported: capture-aware evaluation reveals variance that random splitting conceals. At depth 6, the bootstrap 95% interval for benign recall spans 0.0059 under random splitting and 0.0346 under capture-aware splitting, and the interval is wider under capture-aware splitting at every depth in the grid.

Not supported: picking a single best depth under capture-aware evaluation. The best and second-best settings are depths 20 and 25, whose macro-F1 intervals overlap almost entirely (0.6860–0.7268 against 0.6841–0.7186). Under random splitting, the intervals at depths 30 and 25 are disjoint, so an optimum is identifiable there but not transferable: it lies at a depth whose benign recall is 0.6030, well below what shallower settings achieve.

5.7 An incidental zero-shot observation

Two draws placed an attack family in the test partition with no corresponding training data, permitting an incidental zero-shot observation. Both times the model sent most of the unseen family's flows to other attack families and relatively few to benign. Recon flows were labelled DoS in 50.4% of cases and benign in 12.0%; Brute Force flows were labelled Spoofing in 28.3% of cases and benign in 16.8%.

Family attribution therefore fails for unseen attack types, but binary detection retains partial effectiveness. For an operator, a misnamed alert is recoverable; a missed one is not. We report this as an observation rather than a result: n = 2, and the draws were incidental rather than designed. A deliberate leave-one-family-out evaluation would establish the effect properly and is the obvious follow-up.

6. CONCLUSIONS

We audited four routine decisions in the evaluation of flow-based IoT intrusion detection on CIC IoT-DIAD 2024. Each is ordinarily made by default rather than by argument, and each proved consequential.

Sampling a fixed number of records per capture file makes a class's apparent prevalence a function of how the corpus happens to be split into files; on this dataset the resulting distribution closely tracked the file-count distribution and was radically different from the true flow distribution. Partitioning records at random rather than by capture inflated multiclass accuracy by roughly 12 points and scored macro-F1 by a fifth, and—less obviously but more importantly—concealed a large increase in variance whose magnitude we can bound only loosely: the ratio of standard deviations over twenty draws has a bootstrap 95% interval of 44.9 to 125.9. Comparing balancing strategies without matching training-set size attributed a data-loss effect to balancing and reversed the sign of the conclusion. In an earlier version of this work, reading feature importance off an attribution ranking with no removal control led us to call redundant the only feature pair whose removal turned out to be costly; the controlled experiment overturned that. Finally, leaving tree depth at the library default gave the lowest minority-class recall of all configurations tested under each of the three partitioning protocols, and adjusting depth recovered much more minority-class performance than tuning the decision threshold.

Two of the findings go beyond methodology. One is that two of the eight families in this dataset each come from a single capture file, which means no experimental design can evaluate them under capture-aware splitting; only more data can remove this limit. The other is that when a family appeared at test time without training exposure, the model put most of its flows into other attack families and not into benign traffic. This suggests that, faced with unfamiliar attacks, binary detection degrades less than family attribution does.

The magnitudes we report come from one corpus, one classifier family and the seed counts given above. We expect the mechanisms to hold more widely, but the figures themselves should be taken as one case and not as constants. None of this means that the published results on this dataset are wrong. Our point is that they measure something narrower than is usually assumed, namely performance on capture sessions the model has already seen, under a class distribution set by the way the files are organised. If that is what a study intends to claim, its results remain valid. If the claim concerns generalisation to new deployments, a capture-aware protocol with reported variance is the minimum requirement, and even then this dataset cannot support per-family conclusions for every family.

The obvious next steps are to replicate this audit across other IoT flow benchmarks, to establish whether the magnitudes reported here are typical, and to convert the incidental zero-shot observation of Section 5.7 into a designed leave-one-family-out evaluation.

DATA AVAILABILITY

The CIC IoT-DIAD 2024 dataset is publicly available from the Canadian Institute for Cybersecurity at https://www.unb.ca/cic/datasets/iot-diad-2024.html and is not redistributed by us. All code required to reproduce the sampling protocol, the seven experiments, and every table and figure in this paper is available in the manuscript-associated GitHub release (v1.1.0) at https://github.com/moh1984/iot-diad-protocol-audit/releases/tag/v1.1.0, including the preliminary per-file sampling analysis discussed in Sections 1 and 5.1.

AUTHOR CONTRIBUTIONS

Mohammad Alkhazaleh conceived the study and wrote the manuscript; Mahmoud Baklizi designed the experimental protocol and performed the analysis; Salameh A. Mjlae verified the results and reviewed the literature; Musab Alzghoul curated the data and validated the reproducibility artefacts; Loui Al Sardy contributed to the design and execution of the revision experiments, including the size-

matched partitioning protocol and the control-pair ablation, and critically reviewed the revised analysis and its interpretation. All authors read and approved the final manuscript.

REFERENCES

[1] Sommer, R., Paxson, V. (2010). Outside the closed world: On using machine learning for network intrusion detection. In 2010 IEEE Symposium on Security and Privacy, Oakland, USA, pp. 305-316. https://doi.org/10.1109/SP.2010.25

[2] Arp, D., Quiring, E., Pendlebury, F., et al. (2022). Dos and don'ts of machine learning in computer security. In 31st USENIX Security Symposium, Boston, USA, pp. 3971-3988. https://www.usenix.org/conference/usenixsecurity22/presentation/arp.

[3] Pendlebury, F., Pierazzi, F., Jordaney, R., Kinder, J., Cavallaro, L. (2019). TESSERACT: Eliminating experimental bias in malware classification across space and time. In 28th USENIX Security Symposium, Santa Clara, USA, pp. 729-746. https://www.usenix.org/conference/usenixsecurity19/presentation/pendlebury.

[4] Rabbani, M., Gui, J., Nejati, F., et al. (2025). Device identification and anomaly detection in IoT environments. IEEE Internet of Things Journal, 12(10): 13625-13643. https://doi.org/10.1109/JIOT.2024.3522863

[5] Koroniotis, N., Moustafa, N., Sitnikova, E., Turnbull, B. (2019). Towards the development of realistic botnet dataset in the Internet of Things networks for forensic analytics: Bot-IoT dataset. Future Generation Computer Systems, 100: 779-796. https://doi.org/10.1016/j.future.2019.05.041

[6] Neto, E.C.P., Dadkhah, S., Ferreira, R., Zohourian, A., Lu, R., Ghorbani, A.A. (2023). CICIoT2023: A real-time dataset and benchmark for large-scale attacks in IoT environment. Sensors, 23(13): 5941. https://doi.org/10.3390/s23135941

[7] Chikezie, C.I., Usman, A.U., David, M., Zubair, S., Ohize, H.O., Ojeniyi, J. (2026). A scientific integrity framework for open-set IoT intrusion detection with device-disjoint splits. Future Internet, 18(6): 287. https://doi.org/10.3390/fi18060287

[8] Antonakakis, M., April, T., Bailey, M., et al. (2017). Understanding the Mirai botnet. In 26th USENIX Security Symposium, Vancouver, Canada, pp. 1093-1110. https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/antonakakis.

[9] Sharafaldin, I., Lashkari, A.H., Ghorbani, A.A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. In 4th International Conference on Information Systems Security and Privacy (ICISSP), Funchal, Portugal, pp. 108-116. https://doi.org/10.5220/0006639801080116

[10] Engelen, G., Rimmer, V., Joosen, W. (2021). Troubleshooting an intrusion detection dataset: The CICIDS2017 case study. IEEE Security and Privacy Workshops (SPW), San Francisco, USA, pp. 7-12. https://doi.org/10.1109/SPW53761.2021.00009

[11] Jain, P., Rathour, A., Sharma, A., Chhabra, G.S. (2025). Bridging explainability and security: An XAI-enhanced hybrid deep learning framework for IoT device identification and attack detection. IEEE Access, 13: 127368-127390. https://doi.org/10.1109/ACCESS.2025.3590159

[12] Hossain, M.A. (2025). Deep learning-based intrusion detection for IoT networks: A scalable and efficient approach. EURASIP Journal on Information Security, 2025: 28. https://doi.org/10.1186/s13635-025-00202-w

[13] Sjögren, G., Regnell, R. (2025). Anomaly detection on edge networks. M.Sc. thesis. Department of Electrical and Information Technology, Lund University, Lund, Sweden, Rep. no. LU/LTH-EIT 2025-1050. https://lup.lub.lu.se/student-papers/record/9188366.

[14] AlFuraih, D., Mhamdi, L., Karar, A.S. (2026). Explainable hybrid CNN-XGBoost framework for multi-class IoT intrusion detection with leakage-aware feature selection. Applied System Innovation, 9(3): 49. https://doi.org/10.3390/asi9030049

[15] Rossow, C., Dietrich, C.J., Grier, C., et al. (2012). Prudent practices for designing malware experiments: Status quo and outlook. In 2012 IEEE Symposium on Security and Privacy, San Francisco, USA, pp. 65-79. https://doi.org/10.1109/SP.2012.14

[16] Roberts, D.R., Bahn, V., Ciuti, S., et al. (2017). Cross-validation strategies for data with temporal, spatial, hierarchical, or phylogenetic structure. Ecography, 40(8): 913-929. https://doi.org/10.1111/ecog.02881

[17] Kaufman, S., Rosset, S., Perlich, C., Stitelman, O. (2012). Leakage in data mining: Formulation, detection, and avoidance. ACM Transactions on Knowledge Discovery from Data, 6(4): 15. https://doi.org/10.1145/2382577.2382579

[18] Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P. (2002). SMOTE: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research, 16: 321-357. https://doi.org/10.1613/jair.953

[19] He, H., Garcia, E.A. (2009). Learning from imbalanced data. IEEE Transactions on Knowledge and Data Engineering, 21(9): 1263-1284. https://doi.org/10.1109/TKDE.2008.239

[20] Davis, J., Goadrich, M. (2006). The relationship between precision-recall and ROC curves. In 23rd International Conference on Machine Learning (ICML), Pittsburgh, USA, pp. 233-240. https://doi.org/10.1145/1143844.1143874

[21] Chicco, D., Jurman, G. (2020). The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. BMC Genomics, 21: 6. https://doi.org/10.1186/s12864-019-6413-7

[22] Lundberg, S.M., Lee, S.I. (2017). A unified approach to interpreting model predictions. In Advances in Neural Information Processing Systems 30 (NeurIPS), Long Beach, USA, pp. 4765-4774. https://doi.org/10.48550/arXiv.1705.07874

[23] Breiman, L. (2001). Random forests. Machine Learning, 45(1): 5-32. https://doi.org/10.1023/A:1010933404324

[24] Strobl, C., Boulesteix, A.L., Zeileis, A., Hothorn, T. (2007). Bias in random forest variable importance measures: Illustrations, sources and a solution. BMC Bioinformatics, 8: 25. https://doi.org/10.1186/1471-2105-8-25

[25] Lundberg, S.M., Erion, G., Chen, H., et al. (2020). From local explanations to global understanding with explainable AI for trees. Nature Machine Intelligence, 2(1): 56-67. https://doi.org/10.1038/s42256-019-0138-9
[26] Niculescu-Mizil, A., Caruana, R. (2005). Predicting good probabilities with supervised learning. In 22nd International Conference on Machine Learning (ICML), Bonn, Germany, pp. 625-632. https://doi.org/10.1145/1102351.1102430
[27] Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research, 7: 1-30. https://www.jmlr.org/papers/v7/demsar06a.html.
[28] Pedregosa, F., Varoquaux, G., Gramfort, A., et al. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12: 2825-2830. https://doi.org/10.5555/1953048.2078195
[29] Probst, P., Wright, M.N., Boulesteix, A.L. (2019). Hyperparameters and tuning strategies for random forest. WIREs Data Mining and Knowledge Discovery, 9(3): e1301. https://doi.org/10.1002/widm.1301

NOMENCLATURE

F1	harmonic mean of precision and recall, dimensionless
n	number of records or of independent draws, dimensionless

Greek symbols

σ	standard deviation across independent draws, dimensionless
τ	decision threshold applied to the predicted attack probability, dimensionless

Subscripts

attack	attack class (any of the seven attack families)
benign	benign class

APPENDIX

Reproducibility

All code, the sampling protocol, and the exact configuration of every experiment are available in the GitHub repository, with the manuscript-associated release identified as version v1.1.0: https://github.com/moh1984/iot-diad-protocol-audit/releases/tag/v1.1.0.

Sampling rule: proportional to true family size with a floor of 2,000 flows per family, random seed 42. Corpus enumeration and sampling are performed in two passes; the first counts rows without loading them. All results were produced with Python 3.12.3, scikit-learn 1.4.2, pandas 2.2.2, numpy 2.2.6, scipy 1.13.0 and shap 0.52.0; the repository pins these versions to support reproduction of the reported values under the recorded environment.

The repository also has a preliminary/ directory with the per-file sampling script and its build summary, which are the artefacts referred to in the main text. They are kept as the audited example and are not recommended as a protocol; the repository README labels them accordingly. The configuration of each experiment is summarised in Appendix Table A1.

Table A1. Configuration of each experiment

Experiment	Trees	Max_Depth	Protocol	Seeds
Port ablation	300	25 and None	Random	15 seeds
Protocol comparison	300	25	All three	20 seeds
Balancing control	300	25	Random	15 seeds
Threshold calibration	300	25	Random (3-way)	42
Depth sweep	200	6–30, None	All three	20 seeds
SHAP attribution	300	25 and None	Random	42
Per-family recall	300	25	Capture-aware	20 seeds

Note: All models use class_weight = “balanced_subsample” and median imputation; all use the 72-feature set described in Section 4.2. SHAP: SHapley Additive exPlanations; max_depth: maximum tree depth, with None denoting unbounded depth.