Beyond JA4+: Flow Statistics vs. TLS Fingerprinting for Encrypted Malware Detection

Márton Pál Lipcsey-Magyar, Attila Ármin Madarász, and Adrian Pekar

Abstract—**The deployment of Encrypted Client Hello (ECH) challenges TLS fingerprinting, a widely used approach for encrypted malware detection, by encrypting the handshake fields these methods rely on. This paper presents a systematic evaluation of flow-based statistical features as a handshake-independent alternative to fingerprinting. Through validation against the official JA4+ implementation, we establish limitations in fingerprinting approaches for this corpus: only 64.9% of malware families possess unique signatures, placing an inherent ceiling on achievable recall in our evaluation. We evaluate flow-level features—packet counts, timing patterns, and size distributions—across 27 experimental configurations on a dataset of 16,542 flows spanning 101 families (59 malware and 42 benign applications). Random Forest classifiers using combined flow statistics and sequential packet length features achieve 98.11% F1-score for binary malware detection with 97.22% recall, substantially exceeding fingerprinting's theoretical recall bound of 64.9%. For fine-grained family identification, we obtain 54.81% macro F1 across 101 classes and 48.71% macro F1 for malware-only attribution, demonstrating that flow-based methods retain meaningful discriminative power where fingerprinting abstains. Across all tasks, Random Forest consistently outperforms neural networks and k-NN, with performance gaps widening in complex multiclass scenarios. These findings highlight flow-based classification as a practical and reproducible approach that can help maintain network security visibility as ECH deployment progresses, showing that behavioral traffic patterns are expected to provide durable signals for detection even as handshake fields become encrypted.**

Index Terms—**JA4+ fingerprints, malware classification, flow statistics, encrypted client hello, TLS fingerprinting, network security**

I. Introduction

The analysis of encrypted network traffic is a critical component of modern network security. With most internet traffic now protected by TLS, the ability to identify malicious activity without relying on payload inspection has become essential. One widely deployed approach is TLS fingerprinting, which leverages the fact that different software applications and malware families often produce distinctive handshake signatures. A recent example is the *JA4+ suite* [1], which derives fingerprints from the Client Hello (*JA4*) and Server Hello (*JA4S*), optionally enriched with metadata such as the Server Name Indication (SNI). By making classification decisions at the very beginning of the connection, such fingerprints enable *early detection* and near real-time blocking.

Despite its utility, TLS fingerprinting faces two fundamental challenges. First, fingerprinting is limited by signature coverage: numerous distinct malware families share identical TLS handshakes, and only a subset are uniquely identifiable [2], [3]. Second, fingerprinting assumes visibility into the plaintext handshake. This assumption is being challenged by the deployment of Encrypted Client Hello (ECH), which encrypts the inner Client Hello while exposing only an outer, potentially generic version. As major browsers adopt ECH and CDNs facilitate rollout, the visibility on which client-side fingerprinting relies is expected to diminish [4], [5]. Together, these trends highlight the need for complementary approaches that are less sensitive to changes in TLS protocol visibility.

Flow-level statistical features offer one such alternative. By focusing on observable characteristics such as packet counts, timing patterns, and size distributions, prior work has shown that encrypted malware traffic can be distinguished from benign traffic without access to payloads or handshake fields [6], [7]. These features remain largely observable under encryption enhancements such as ECH, making them a more durable basis for traffic analysis.

In this paper, we systematically evaluate flow-level statistical features as an ECH-resilient complement to TLS fingerprinting. Using a validated malware dataset containing 16,542 flows across 101 families, we demonstrate that flow-based classifiers provide robust detection in binary tasks and meaningful discriminative power for multi-family attribution, substantially exceeding the theoretical recall limits of TLS fingerprinting.

Our contributions are as follows:

- A systematic evaluation of flow-based features for ECH-resilient malware detection, demonstrating that flow statistics achieve 98.11% F1 in binary detection without requiring handshake inspection—a capability expected to persist as ECH adoption renders traditional fingerprinting increasingly limited.
- A quantitative comparison framework establishing theoretical upper bounds for TLS fingerprinting (precision $\approx$99.62%, recall $\leq$64.9%, F1 $\leq$78.6%) and demonstrating that flow-based methods exceed fingerprinting's recall bound by 32+ percentage points (97.22% vs. 64.9%).
- Feature importance analysis across all three tasks revealing task-dependent patterns—packet sizes dominate binary detection while temporal features dominate family attribution—with all top features remaining ECH-resilient

M. P. Lipcsey-Magyar, A. Á. Madarász, and A. Pekar are with the Budapest University of Technology and Economics, Hungary
(e-mail: lipcsey-magyarmartonpal@edu.bme.hu, madarasz.attila@edu.bme.hu, apekar@hit.bme.hu).
M. P. Lipcsey-Magyar, A. Á. Madarász, and A. Pekarare also with HUN-REN Office for Supported Research Groups, Hungary.
A. Pekar is also with CUJO LLC, Budapest, Hungary.

DOI: 10.36244/ICJ.2026.1.4

and cross-validation (5-fold, F1 std <0.002 for binary) confirming result stability.
- A validated pipeline where JA4/JA4S extraction conforms to the official JA4+ suite (99.98% flow matching; 100% hash conformance) and a feature-rich dataset of 16,542 flows across 101 families with 89 properties enabling reproducible research.

The remainder of this paper is organized as follows. Section II reviews related work in encrypted traffic analysis and TLS fingerprinting. Section III describes our methodology, including dataset validation against official JA4+ implementations, flow feature extraction, and the classification framework across three tasks and three algorithms. Section IV presents experimental results for binary malware detection, multiclass family identification, and comparative analysis with TLS fingerprinting baselines. Section V discusses operational implications for ECH-resilient detection, explains performance patterns, and identifies limitations and future research directions. Section VI concludes the paper.

II. Related Work

Network security in encrypted environments has driven extensive research into methods for identifying malicious traffic without payload inspection. We organize existing work into three primary categories: TLS fingerprinting approaches, statistical flow-based detection methods, and hybrid contextual techniques.

A. TLS Fingerprinting Methods

TLS fingerprinting leverages unique signatures generated during the TLS handshake process for encrypted traffic analysis. The seminal JA3 method [8] extracts fingerprints from Client Hello messages by hashing TLS version, cipher suites, extensions, elliptic curves, and formats. Complementing this, JA3S fingerprints capture Server Hello responses, enabling bidirectional analysis that significantly improves malware detection accuracy over unidirectional approaches.

Recent advances culminated in the JA4+ suite [1], [9], which addresses several limitations of JA3 through improved hash stability and expanded coverage. Matoušek et al. [10] conducted a comprehensive evaluation of JA4+ fingerprints across 64 malware families, reporting that combining JA4, JA4S, and SNI achieves 87% uniqueness with approximately 80% family coverage on their dataset. However, they also report residual overlap between malware and benign fingerprints (e.g., certificate hashes) and cases where distinct families share identical handshake signatures.

The reliability challenges are further exposed by Matoušek et al. [11] in their analysis of mobile applications, where JA3 fingerprints alone prove insufficient due to high collision rates among different applications. Similarly, Siwakoti and Rawat [12] highlight that while machine learning can extend beyond known fingerprint signatures, achieving 96.4% accuracy in their evaluation, the fundamental coverage problem persists—only a fraction of malware families possess unique TLS signatures.

Beyond JA3/JA4-style techniques, recent work explores broader fingerprinting strategies. Ede et al. [13] proposed *FLOWPRINT*, a semi-supervised fingerprinting framework that demonstrated how mobile apps with distinct behaviors may nonetheless collide under identical TLS handshakes, underscoring coverage limitations. On the server side, Theofanous et al. [14] presented *Fingerprinting the Shadows*, which unmasked malicious servers behind CDNs and proxies. Their work shows that while advanced server fingerprinting remains feasible, visibility is increasingly constrained by encrypted handshake fields and middleboxes.

B. Statistical Flow-Based Detection

An alternative paradigm leverages statistical characteristics of network flows, offering potential resilience to encryption enhancements. Anderson and McGrew [15] pioneered this approach by combining TLS metadata with DNS and HTTP contextual features, achieving 99.978% accuracy on their dataset via contextual correlation across protocols. Their "data omnia" philosophy demonstrates that unencrypted metadata can effectively identify malicious encrypted communications.

Piskozub et al. [16] advanced purely flow-based detection with MalAlert, which aggregates flows into "flowsets" and extracts 441 statistical features for malware classification. Their approach achieves 90% F1-score for malware type identification in their experiments while maintaining privacy through byte-level statistics rather than content inspection. Crucially, this method operates independently of TLS handshake visibility.

Yeo et al. [17] explored deep learning approaches using convolutional neural networks on 35 flow statistical features, achieving over 85% accuracy across multiple malware families in their evaluation. Their comparison with traditional machine learning methods (Random Forest, SVM) showed that both CNN and Random Forest exceed 93% performance across all evaluation metrics on their dataset.

Barut et al. [7] provided a comprehensive performance study comparing machine learning and deep learning approaches on flow features, uniquely addressing computational efficiency alongside accuracy. Their analysis revealed that traditional machine learning methods often outperform deep learning while requiring fewer computational resources, with acceleration libraries providing up to 68.6× speedup.

Building on these foundations, Fu et al. [18] recently demonstrated that unknown encrypted malicious traffic can be detected in real time using flow statistics and side-channel features, without relying on handshake visibility. Collectively, these works show that flow-based methods can achieve strong accuracy with modest computational overhead, making them practical for real-time monitoring deployments.

C. Hybrid and Advanced Approaches

Recent work has explored techniques that combine multiple signal sources or advanced pattern recognition. Kim et al. [19] revisited Markov chain-based fingerprinting as an alternative to ClientHello parsing, achieving 88.1% accuracy for malware family classification in their experiments. Their

approach offers advantages including resilience to ECH and generalizability for approximate matching.

Yu and Won [20] provide a comprehensive taxonomy of encrypted traffic fingerprinting methods, categorizing approaches into fingerprint collection techniques and AI-based classification methods. Their analysis identifies the growing need for hybrid approaches that combine the speed of traditional fingerprinting with the accuracy and content detection capabilities of machine learning techniques. However, hybrid methods remain less mature than either pure fingerprinting or flow-based statistics, with limited validation in operational settings.

D. Limitations and Research Gaps

Despite significant progress, existing work exhibits several limitations that motivate our approach. *Reproducibility gaps* affect fingerprinting validation: few studies validate extraction against official JA4+ implementations, and processing pipelines vary across research groups, limiting reproducibility. *Evaluation methodology gaps* for fingerprinting are widespread: most studies lack train/test splits, systematic comparison frameworks with theoretical bounds, and fail to report coverage or abstention rates.

The *ECH constraint* challenges handshake-dependent methods: with major browsers implementing ECH support and CDN providers beginning rollout, methods that inspect ClientHello/ServerHello fields may lose visibility into the attributes they depend on. This motivates approaches that do not require handshake field inspection. Finally, *metric limitations* affect multiclass evaluation: many studies report accuracy or weighted averages rather than macro-averaged metrics that appropriately handle long-tail class distributions common in malware datasets.

In this work, we directly address these gaps through a systematic evaluation of ECH-resilient flow features, a principled comparison against TLS fingerprinting using both theoretical bounds and empirical results, and a unified dataset foundation that establishes clear baselines for future research.

III. METHODOLOGY

Our research employs the public dataset provided by Matoušek et al. [10], which contains authenticated network traces for both malware communications and benign mobile and desktop applications. The dataset is distributed in two formats: labeled CSV files containing pre-extracted TLS fingerprinting metrics and PCAP network traces. The dataset authors noted that the public repository contains a subset of PCAP files due to size limitations, while the CSV files represent the complete experimental record. Through direct communication with the authors, we obtained the full PCAP collection, ensuring our analysis uses the complete dataset for comprehensive evaluation of ECH-resilient classification approaches.

A. Flow Feature Extraction and Validation

Our analysis required comprehensive flow-level statistical features that were not available in the author-provided CSV files, which contained exclusively TLS fingerprinting metrics. To obtain the necessary features for ECH-resilient classification, we extracted flow records directly from the complete PCAP collection using NFStream [21], a high-performance network flow analysis framework. This extraction process enabled us to capture bidirectional flow characteristics that remain observable under encrypted communication channels, including those protected by ECH deployment.

1) NFStream Configuration and Feature Extraction: We configured NFStream to extract bidirectional flows using standard 5-tuple identification (source IP, destination IP, source port, destination port, protocol) with industry-standard timeout parameters: 120 seconds for idle flows and 1800 seconds for active connections. The framework extracted comprehensive flow statistics, providing coarse- and fine-grained traffic signatures.

To enable direct comparison with existing TLS fingerprinting approaches, we developed an NFStream plugin implementing the official JA4+ specification [1]. This plugin generates both JA4 (client) and JA4S (server) fingerprints in strict conformance with the published standard, ensuring reproducibility and enabling systematic comparison between fingerprinting and flow-based approaches.

2) Validation Against Established Baselines: To validate our NFStream extraction methodology, we performed systematic comparative analysis using the author-provided datasets. As the initial comparison revealed several important variations in data formatting and filtering approaches that made a direct, one-to-one analysis challenging, we developed a *unified filtering pipeline* that ensures consistent, reproducible comparisons between the author-provided CSVs and our NFStream-generated data. Our pipeline implements four systematic filtering steps:

1) Removing entries with missing JA4 hashes, as our research focuses exclusively on TLS connections.
2) Filtering flows labeled as "Unknown" applications to ensure classification accuracy.
3) Applying the author-provided SNI-based filtering list to remove advertising and tracking traffic that introduces classification noise.
4) Removing families with only single samples.

The SNI filtering list proved particularly critical, as it included not only advertising services but also ubiquitous domains that appear across multiple malware families and benign applications. These overlapping domains can yield identical fingerprints that significantly impact classification performance, making their removal essential for accurate evaluation.

3) Comparative Analysis of Fingerprinting Metrics: Table I presents our comparative analysis of key TLS fingerprinting metrics after applying our unified filtering pipeline. We evaluate four critical metrics following the definitions from [10]:

- *Coverage*: The percentage of flows for which a specific fingerprint type can be extracted. JA4 requires only the client hello message, while JA4S additionally requires the server response, making it more susceptible to incomplete handshakes.

TABLE I: TLS Fingerprinting Performance After Unified Filtering

Metric	Author CSV (Filtered)	NFStream (Filtered)
Dataset Characteristics		
Total Flows	15,157	16,566
JA4 Coverage	15,157 (100%)	16,566 (100%)
JA4S Coverage	13,934 (91.9%)	7,821 (47.2%)
SNI Coverage	14,732 (97.2%)	15,144 (91.4%)
JA4+JA4S+SNI Performance		
Total Unique Fingerprints	2,423	1,363
Uniqueness	89.4%	89.9%
Malware Family Coverage	75.0%	64.9%
Application Coverage	94.2%	93.1%
Malware-Benign Overlap	1.23%	0.38%

- *Uniqueness*: The percentage of fingerprints that uniquely identify a single malware family or application, indicating the discriminative power of the fingerprinting method.
- *Family Coverage*: The percentage of malware families or applications that possess at least one unique fingerprint, representing the method's ability to identify distinct behavioral patterns.
- *Overlap*: The percentage of fingerprints shared between malware and benign traffic, directly impacting false positive rates and classification ambiguity.

The comparative analysis reveals a critical divergence in JA4S coverage between our NFStream extraction (47.2%) and the author-provided CSVs (91.9%). Despite this substantial difference in server-side fingerprint availability, the metrics most critical for security applications demonstrate remarkable consistency. Uniqueness remains high at approximately 90% in both approaches, application coverage exceeds 93%, and crucially, the malware-benign overlap remains minimal (0.38–1.23%). These consistent patterns suggest that while the two extraction methods capture different numbers of complete handshakes, the fundamental discriminative power of TLS fingerprinting is preserved when combining JA4, JA4S, and SNI attributes.

4) Implementation Validation Against Official JA4+ Suite: The significant discrepancy in JA4S coverage between our NFStream extraction (47.2%) and the author-provided CSVs (91.9%) necessitated independent validation to establish ground truth. We processed the complete PCAP collection using the official JA4+ suite implementation, generating authoritative fingerprints. This validation serves two critical purposes: verifying the correctness of our extraction methodology and understanding the source of coverage discrepancies.

Table II presents the conformance analysis results. We evaluated two key metrics: *Flow Match Rate* (the percentage of flows identified by the official tool that were also present in each dataset) and *Fingerprint Conformance* (the percentage of matched flows for which the JA4/JA4S hashes were identical to those generated by the official implementation).

The validation results are conclusive and reveal important methodological insights. Our NFStream pipeline successfully matches and processes 99.98% of flows identified by the official JA4+ tool, with every extracted fingerprint conforming perfectly to the official specification (100% conformance for both JA4 and JA4S). This complete conformance validates the correctness of our implementation and provides high confidence in subsequent analyses.

TABLE II: Conformance with Official JA4+ Suite Implementation

Metric	Author CSV	NFStream
Total Flows in Dataset	33,589	163,897
Official Tool Flows (PCAPs)	43,132	43,345
Matched Flows (5-tuple)	33,577	43,337
Flow Match Rate	77.85%	99.98%
Among Matched Flows:		
JA4 Conformance	30,077/33,577 (89.58%)	32,787/32,787 (100%)
JA4S Conformance	15,924/31,846 (50.00%)	15,470/15,470 (100%)

In contrast, the author-provided CSVs exhibit significant discrepancies: missing over 22% of TLS flows present in the PCAPs and achieving only 50% JA4S conformance among matched flows. This strongly suggests that the high JA4S coverage (91.9%) reported in the author-curated CSVs results from a non-standard extraction process rather than actual server response availability in the network traces. The discrepancy likely stems from different interpretations of incomplete handshakes or alternative fingerprint computation methods not aligned with the official specification. Thus, earlier reports of ∼92% JA4S coverage appear inflated due to non-standard extraction pipelines, whereas our validation against the official JA4+ suite indicates that the true coverage in real PCAP traces is closer to 47%.

5) Implications for Flow-Based Analysis: These validation findings have several important implications for our research:

1) *Methodological Rigor*: The complete conformance with official JA4+ specifications ensures that our comparisons between fingerprinting and flow-based approaches are based on accurate, standardized implementations rather than potentially biased extraction methods.
2) *Coverage Limitations*: The actual JA4S coverage of 47.2% in real network traces highlights a fundamental limitation of server-dependent fingerprinting methods. In contrast, our flow-based features can be extracted from any TLS connection regardless of handshake completion.
3) *Reproducibility*: Our NFStream-based extraction pipeline, validated against the official implementation, provides a reproducible foundation for future research. *All extraction code and processed datasets are made available to ensure scientific reproducibility [22].*
4) *Performance Baselines*: The consistent performance patterns across key security metrics (uniqueness 90%, minimal overlap 0.38%) despite different extraction methods demonstrate that our dataset captures the essential characteristics needed for meaningful classification experiments.

The systematic filtering pipeline ensures that all subsequent comparisons between fingerprinting and flow-based approaches are conducted on a consistent, reproducible basis.

While absolute coverage values vary between extraction methods, the fundamental ability to discriminate between malware and benign traffic remains intact, providing a solid foundation for evaluating ECH-resilient alternatives to traditional TLS fingerprinting.

B. Feature Engineering

Our feature engineering approach leverages three complementary sets of traffic characteristics that remain observable under encryption and, critically, are resilient to ECH deployment. Unlike TLS fingerprinting methods that rely on plaintext handshake fields susceptible to privacy-enhancing technologies, our features capture fundamental communication patterns that persist regardless of encryption enhancements.

1) Core Flow Statistics: We extracted 33 bidirectional flow metrics from NFStream, capturing fundamental behavioral characteristics of network communications. These features are computed across three directional perspectives to capture both asymmetric communication patterns and aggregate behavior:

- *Source-to-destination (src2dst)*: Metrics computed exclusively for packets traveling from the flow initiator to the responder, capturing client request patterns and command transmission characteristics.
- *Destination-to-source (dst2src)*: Metrics for response traffic, revealing server behavior patterns and data exfiltration characteristics particularly relevant for malware detection.
- *Bidirectional*: Cumulative aggregation of both directions, where values represent the sum of src2dst and dst2src metrics (*e.g.*, if src2dst contains 10 packets and dst2src contains 15 packets, the bidirectional packet count equals 25).

The comprehensive feature set encompasses:

- *Volumetric metrics* (9 features): Packet counts and byte volumes computed separately for each directional perspective, capturing traffic intensity and data transfer patterns characteristic of different malware families.
- *Temporal characteristics* (3 features): Flow duration in milliseconds for each perspective, revealing communication session patterns and distinguishing between ephemeral connections and persistent command-and-control channels.
- *Packet size distributions* (12 features): Minimum, mean, standard deviation, and maximum packet sizes for each directional perspective. These statistical moments capture protocol-specific behaviors that remain consistent even under encryption, such as characteristic message sizes in malware communication protocols.
- *Packet inter-arrival times (PIAT)* (12 features): Minimum, mean, standard deviation, and maximum inter-arrival times in milliseconds for each perspective. PIAT distributions reveal timing patterns indicative of automated malware behavior versus human-driven benign applications, providing crucial discriminative signals for classification.

These 33 features capture *macro-level communication patterns* that persist regardless of encryption enhancements. The bidirectional perspective provides both directional asymmetry information (through comparison with unidirectional metrics) and aggregate flow behavior, enabling robust traffic classification even when individual packet contents are completely opaque.

2) Sequential Packet Length (SPL) Features: To capture *micro-level communication patterns* that characterize specific application protocols, we extracted Sequential Packet Length (SPL) features consisting of the sizes of the first 25 packets in each flow, ordered by arrival time. This approach is motivated by the observation that many application protocols, including malware command-and-control communications, exhibit characteristic packet size sequences during connection establishment and initial data exchange.

The 25-packet window was selected based on empirical analysis showing that most distinctive protocol behaviors manifest within this initial exchange, while longer sequences provide diminishing returns for classification accuracy.

3) Combined Feature Set: We constructed a hybrid feature representation incorporating both the 33 core flow statistics and 25 SPL values, resulting in a 58-dimensional feature vector. This combined approach aims to leverage complementary information from both feature types:

- *Macro-level patterns* from flow statistics capture overall communication behavior, session characteristics, and traffic intensity patterns that distinguish malware families with different operational profiles.
- *Micro-level signatures* from SPL sequences identify specific protocol implementations and message exchange patterns unique to particular malware variants or benign applications.

The synergy between these feature types addresses limitations of each individual approach: flow statistics may miss subtle protocol variations while SPL features alone may not capture broader behavioral patterns. The combined representation provides comprehensive traffic profiles that remain robust under current and future encryption enhancements, including ECH deployment.

C. Classification Framework

We evaluated three machine learning algorithms across three distinct classification tasks to comprehensively assess the effectiveness of flow-based features for ECH-resilient malware detection.

1) Classification Tasks: We designed three classification tasks to address different operational requirements in malware detection systems:

1) *Binary Classification*: Distinguishing malware from benign traffic, representing the most critical security task where minimizing false negatives is paramount for preventing successful attacks.
2) *Full Multiclass Classification*: Identifying specific families among all classes (malware families and benign applications), enabling fine-grained threat attribution and targeted response strategies.
3) *Malware-Only Multiclass Classification*: Discriminating between malware families exclusively, isolating the challenge of malware family attribution without the simplifying presence of benign traffic patterns.

Throughout this paper, we use the term *class* to denote either a malware family or a benign application family, depending on task context.

2) Machine Learning Algorithms: We selected three complementary algorithms representing different learning paradigms and operational trade-offs:

Random Forest (RF): We configured ensemble classifiers with 300 decision trees, a parameter selected through empirical convergence analysis showing negligible improvement beyond this threshold. The maximum tree depth is set to 15 for binary classification and 20 for multiclass tasks, providing sufficient model capacity while preventing overfitting. Regularization parameters include minimum samples split of 5 and minimum samples leaf of 2, determined through preliminary experiments to prevent memorization while maintaining discriminative power. To address class imbalance inherent in malware datasets, we employ balanced class weighting that inversely weights classes proportional to their frequency. This ensures minority malware families receive appropriate attention during training, crucial for detecting rare threats.

Neural Networks (NN): We implemented fully-connected architectures with batch normalization and dropout regularization, tailored to task complexity. For binary classification, we employ a funnel architecture (64→32→16→1 neurons) that progressively compresses feature representations toward the decision boundary. Multiclass tasks utilize a wider architecture (256→128→64→*num_classes*) to accommodate the increased complexity of distinguishing between up to 101 distinct traffic patterns. ReLU activations after batch normalization address vanishing gradient problems while accelerating convergence. Dropout rates of 0.3 and 0.2 at different layers prevent neuron co-adaptation, with the Adam optimizer (learning rate 10^{-3}) providing adaptive per-parameter learning rates crucial for features with different scales. Early stopping with patience of 3 epochs prevents overfitting while ensuring convergence, monitoring validation loss to restore optimal weights.

FAISS k-Nearest Neighbors (k-NN): We implemented Facebook AI Similarity Search (FAISS) [23] for scalable nearest neighbor classification, using `IndexFlatIP` (inner product similarity) on L2-normalized features to approximate cosine similarity. This choice enables efficient similarity search in high-dimensional feature spaces while maintaining geometric interpretability. We selected $k = 7$ for binary classification and $k = 5$ for multiclass tasks, representing an empirically-determined trade-off between local smoothness and robustness to outliers. Odd values prevent tied votes in classification decisions. These parameters were optimized based on expected class density in the feature space, with larger k for binary classification increasing stability in the simpler two-class problem while smaller k for multiclass tasks preserves local neighborhood structure necessary for fine-grained discrimination.

3) Training and Evaluation Methodology: All models were trained using stratified 80-20 train-test splits with fixed random seeds (42) to ensure complete reproducibility. Stratification maintains proportional representation of all classes in both training and test sets, crucial for unbiased performance estimation in imbalanced datasets.

Building upon our initial filtering (*cf.* Section III-A2) that removed singleton samples, we applied an additional minimum support threshold of 5 samples per family. This threshold is mathematically necessary for stratified sampling: classes with fewer than 5 samples would yield zero test samples after the 80-20 split ($\lfloor 4 \times 0.2 \rfloor = 0$), violating the requirement for representative evaluation. With exactly 5 samples, we guarantee at least one test sample per class ($\lfloor 5 \times 0.2 \rfloor = 1$), ensuring all families are represented in both training and evaluation sets.

This additional filtering eliminated 8 underrepresented families (24 samples total), yielding a final dataset of 16,542 records spanning 101 unique families, with 59 malware families and 42 benign application families.

4) Evaluation Metrics: We report standard classification metrics including *accuracy*, *precision*, *recall*, and *F1-score*. For multiclass tasks, we report macro-averaged metrics to avoid bias toward majority classes, ensuring that performance on rare malware families is appropriately weighted in aggregate scores. This is particularly important given that rare malware variants often represent emerging threats requiring immediate detection capability.

Additionally, we report *ROC-AUC* (Receiver Operating Characteristic - Area Under Curve) scores to characterize classifier confidence and decision robustness beyond threshold-dependent metrics. For binary classification, ROC-AUC measures the probability that a randomly chosen positive sample ranks higher than a randomly chosen negative sample. For multiclass tasks, we compute macro-averaged ROC-AUC using the One-vs-Rest strategy, providing insight into per-class discriminability.

5) Cross-Validation for Stability Assessment: While our primary results use a single stratified 80-20 split for direct comparability with prior work, we additionally perform 5-fold stratified cross-validation to assess model stability and transferability. Cross-validation provides variance estimates that quantify how sensitive performance is to the particular train-test partition, addressing concerns about result generalizability.

We employ `StratifiedKFold` with 5 folds and a fixed random seed (42) to ensure reproducibility. For each fold, models are trained on 80% of the data and evaluated on the held-out 20%, with metrics aggregated as mean ± standard deviation across folds. This protocol enables assessment of whether the single-split results are representative or anomalous, and provides confidence intervals for deployment planning.

D. Comparison Methodology with TLS Fingerprinting

To establish the relative merits of our ECH-resilient approach against traditional TLS fingerprinting, we developed a systematic comparison framework addressing the fundamental differences between deterministic fingerprint matching and probabilistic machine learning classification.

1) Deriving Comparable Metrics: TLS fingerprinting is a deterministic lookup: a flow either matches a known fingerprint or it does not. In contrast, our flow-based approach is probabilistic classification trained and evaluated on stratified splits. To enable a fair, apples-to-apples comparison

without overclaiming, we derive optimistic upper bounds for fingerprinting performance from two corpus-level quantities measured on our NFStream-cleaned dataset: *(i)* the percentage of fingerprint keys shared between malware and benign traffic ("overlap"), and *(ii)* the fraction of malware families that have at least one identifying fingerprint ("malware family coverage").

We derive optimistic upper bounds under the assumption that fingerprint keys are equally likely across flows:

- $P_{\max} = 1 - \text{overlap}$,
- $R_{\max} = \text{family coverage}$, and
- $F1_{\max} = \frac{2 \cdot P_{\max} \cdot R_{\max}}{P_{\max} + R_{\max}}$.

Using our NFStream-cleaned dataset (Table I), overlap is 0.38% and malware family coverage is 64.9%, yielding $P_{\max} \approx 99.62\%$, $R_{\max} \approx 64.90\%$, and $F1_{\max} \approx 78.6\%$.

These bounds favor fingerprinting and represent optimistic estimates: overlap is computed on unique fingerprints, not weighted by flow frequency; if shared fingerprints are high-frequency, real false positive rates could be higher and precision lower. Similarly, the recall bound ignores per-flow extraction failures and incomplete handshakes that limit coverage in practice.

IV. Experimental Results

We present a comprehensive evaluation of flow-based statistical features for handshake-independent malware detection, systematically exploring 27 configurations across three classification tasks, three feature sets, and three machine learning algorithms, demonstrating the viability of flow-based approaches as practical complements to traditional TLS fingerprinting.

A. Binary Classification: Malware Detection

Binary classification represents the most critical security task: distinguishing malware from benign traffic. This fundamental capability determines whether a network security system can identify threats regardless of their specific family or variant. Table III presents comprehensive results across all feature sets and models.

TABLE III: Binary Classification Performance (Malware vs. Benign). Random Forest with combined features achieves the best performance (F1=0.9811), with all configurations exceeding 92.5% F1-score.

Feature Set	Model	Accuracy	Precision	Recall	F1-Score
Core	NN	0.8870	0.9530	0.9000	0.9258
	RF	0.9655	0.9866	0.9691	0.9778
	FAISS	0.9099	0.9338	0.9525	0.9431
SPL	NN	0.8876	0.9305	0.9255	0.9280
	RF	0.9665	0.9874	0.9695	0.9784
	FAISS	0.8849	0.9102	0.9464	0.9279
Combined	NN	0.9003	0.9439	0.9278	0.9358
	RF	**0.9707**	**0.9902**	**0.9722**	**0.9811**
	FAISS	0.8906	0.9148	0.9487	0.9314

Random Forest consistently achieves superior performance across all feature combinations, with the combined feature set reaching 97.07% accuracy and 98.11% F1-score. Recall rates across all configurations range from 90.00% to 97.22%.

Several key patterns emerge from the binary classification results. Core flow statistics achieve 97.78% F1-score with Random Forest, while SPL features achieve 97.84%—nearly identical performance when used independently. The combined feature set reaches 98.11% F1-score (+0.33 percentage points over core features alone).

Random Forest outperforms neural networks by 4.5–5.2 percentage points in F1-score across all feature configurations. Even the lowest-scoring configuration—FAISS k-NN with SPL features—achieves 92.79% F1-score, and all nine binary configurations achieve F1-scores above 92.5%.

Precision-recall analysis reveals distinct model characteristics: Random Forest models maintain high precision (98.66–99.02%) and the highest recall (96.91–97.22%), while neural networks show higher variance with precision ranging from 93.05% to 95.30% and recall from 90.00% to 92.78%. FAISS k-NN achieves strong recall (94.64–95.25%) at the cost of lower precision (91.02–93.38%).

ROC-AUC Analysis: To characterize classifier confidence beyond threshold-dependent metrics, we computed ROC-AUC scores for all binary classification configurations. Table IV presents the results, demonstrating that Random Forest achieves near-perfect discrimination with ROC-AUC scores exceeding 0.99 across all feature sets.

TABLE IV: ROC-AUC Scores for Binary Classification. Random Forest achieves near-perfect discrimination (0.99+) across all feature sets, indicating robust ranking of malware above benign traffic regardless of threshold selection.

Model	Core	SPL	Combined
Neural Network	0.9422	0.9288	0.9420
Random Forest	**0.9933**	**0.9947**	**0.9949**
FAISS k-NN	0.9479	0.9212	0.9270

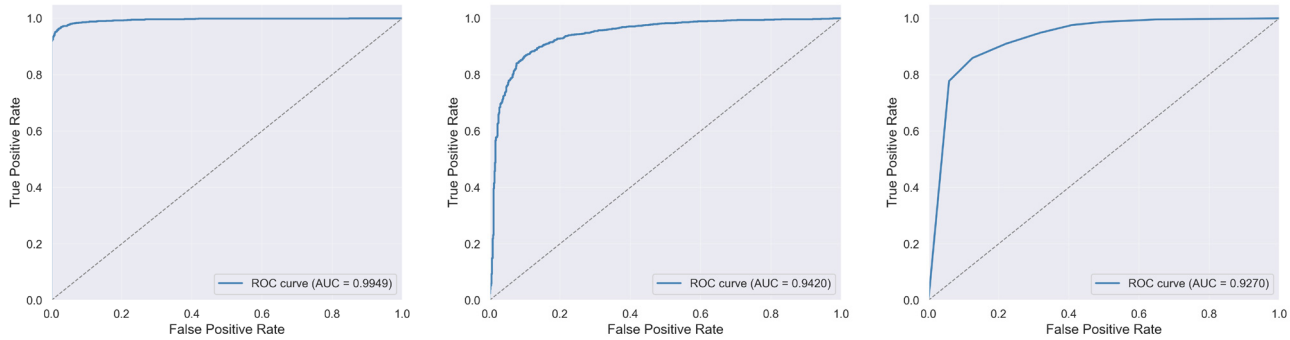
The ROC-AUC results reinforce the F1-score findings: Random Forest demonstrates exceptional discriminative ability, correctly ranking malware flows above benign flows with 99.49% probability when using combined features (99.61% in 5-fold CV). This high ROC-AUC indicates robust performance across all classification thresholds, not merely at the default 0.5 boundary. Fig. 1 visualizes the ROC curves for all three models using combined features, illustrating the near-perfect discrimination achieved by Random Forest.

B. Multiclass Classification: Complete Family Identification

The full multiclass task—distinguishing between all 101 classes (59 malware families and 42 benign application families)—represents the most challenging classification scenario, requiring models to capture subtle differences between semantically similar traffic patterns across 101 classes. Table V presents the performance across all experimental configurations.

Random Forest with combined features achieves the highest performance at 61.62% accuracy and 54.81% F1-score. This represents a substantial drop relative to binary classification (98.11% vs. 54.81% F1-score).

Fig. 2 shows the normalized confusion matrix for the best multiclass configuration across 101 families. The matrix



(a) Random Forest (b) Neural Network (c) FAISS k-NN

Fig. 1: ROC curves for binary classification using combined features. Random Forest achieves near-perfect discrimination, substantially outperforming Neural Network and FAISS k-NN across all decision thresholds.

TABLE V: Full Multiclass Classification Performance (101 Families). Performance drops substantially from binary detection, with Random Forest (combined features) achieving 54.81% macro F1—still meaningful for family attribution.

Feature Set	Model	Accuracy	Precision	Recall	F1-Score
Core	NN	0.4113	0.2685	0.2254	0.2053
	RF	0.5966	0.5191	0.5559	0.5239
	FAISS	0.4642	0.3966	0.3772	0.3743
SPL	NN	0.4092	0.2780	0.2447	0.2353
	RF	0.5561	0.5066	0.5055	0.4923
	FAISS	0.4180	0.3180	0.2981	0.2948
Combined	NN	0.4527	0.3482	0.2895	0.2889
	RF	**0.6162**	**0.5526**	**0.5738**	**0.5481**
	FAISS	0.4397	0.3794	0.3387	0.3430

exhibits a strong diagonal pattern with off-diagonal confusion clusters where related families are misclassified into each other.

Per-class analysis of the same model reveals that perfect accuracies are concentrated in classes with very small test support (≤ 12 flows), while several moderate-support families exhibit systematic confusion (e.g., *mega-wins-slot*: 31 test samples, 0% accuracy; *jelly-connect*: 27, 0%; *njrat*: 15, 0%).

Several important patterns emerge from the multiclass results. Core flow statistics outperform SPL features in isolation (52.39% vs. 49.23% macro F1 with Random Forest; +3.12 percentage points), contrasting with binary classification where the two feature types achieve nearly identical performance (97.78% vs. 97.84%). The combined feature set improves macro F1 by 2.42 percentage points over core flow statistics (54.81% vs. 52.39%).

Neural networks achieve only 28.89% macro F1 with combined features compared to Random Forest's 54.81% (gap: 25.92 percentage points). The dataset exhibits substantial imbalance (median 91 samples per family; min 5, max 1845; 18 classes with < 20 samples and 32 with < 50). Unlike binary classification where models maintain balanced precision and recall, multiclass results show significant divergence: Random Forest with combined features achieves 55.26% precision and 57.38% recall, while neural networks exhibit both low precision (34.82%) and low recall (28.95%).

Performance scaling analysis reveals substantial degradation from binary to multiclass tasks. Random Forest degrades by 43.30 percentage points (from 98.11% to 54.81% macro F1), FAISS k-NN by 58.84 percentage points (from 93.14% to 34.30%), and neural networks by 64.69 percentage points (from 93.58% to 28.89%).

ROC-AUC Analysis: Despite the lower F1-scores in multiclass classification, ROC-AUC scores remain high, suggesting strong class separability at the probability level on this dataset. Table VI presents ROC-AUC scores for the full multiclass task (101 classes), computed using macro-averaged One-vs-Rest (OvR) scoring.

TABLE VI: ROC-AUC Scores for Multiclass Classification (101 Classes). Despite lower F1-scores, Random Forest achieves 0.9768 ROC-AUC, indicating strong per-class discriminability.

Model	Core	SPL	Combined
Neural Network	0.9527	0.9521	0.9629
Random Forest	**0.9761**	**0.9562**	**0.9768**
FAISS k-NN	0.8113	0.7518	0.7714

The high ROC-AUC scores (0.9768 for Random Forest) contrast with the moderate F1-scores (0.5481), suggesting that the classifier assigns useful probability rankings even when hard predictions are incorrect. This is consistent with misclassifications occurring among closely related families with similar probabilities, rather than confident errors. Fig. 3 visualizes the macro-averaged and micro-averaged ROC curves for all three models.

C. Malware-Only Multiclass Classification

To isolate the challenge of discriminating between malware families without the simplifying presence of benign traffic, we evaluated classification performance across the 59 malware families exclusively. Table VII presents the results for this focused classification task.

The malware-only classification achieves a maximum F1-score of 48.71% with Random Forest on combined features.

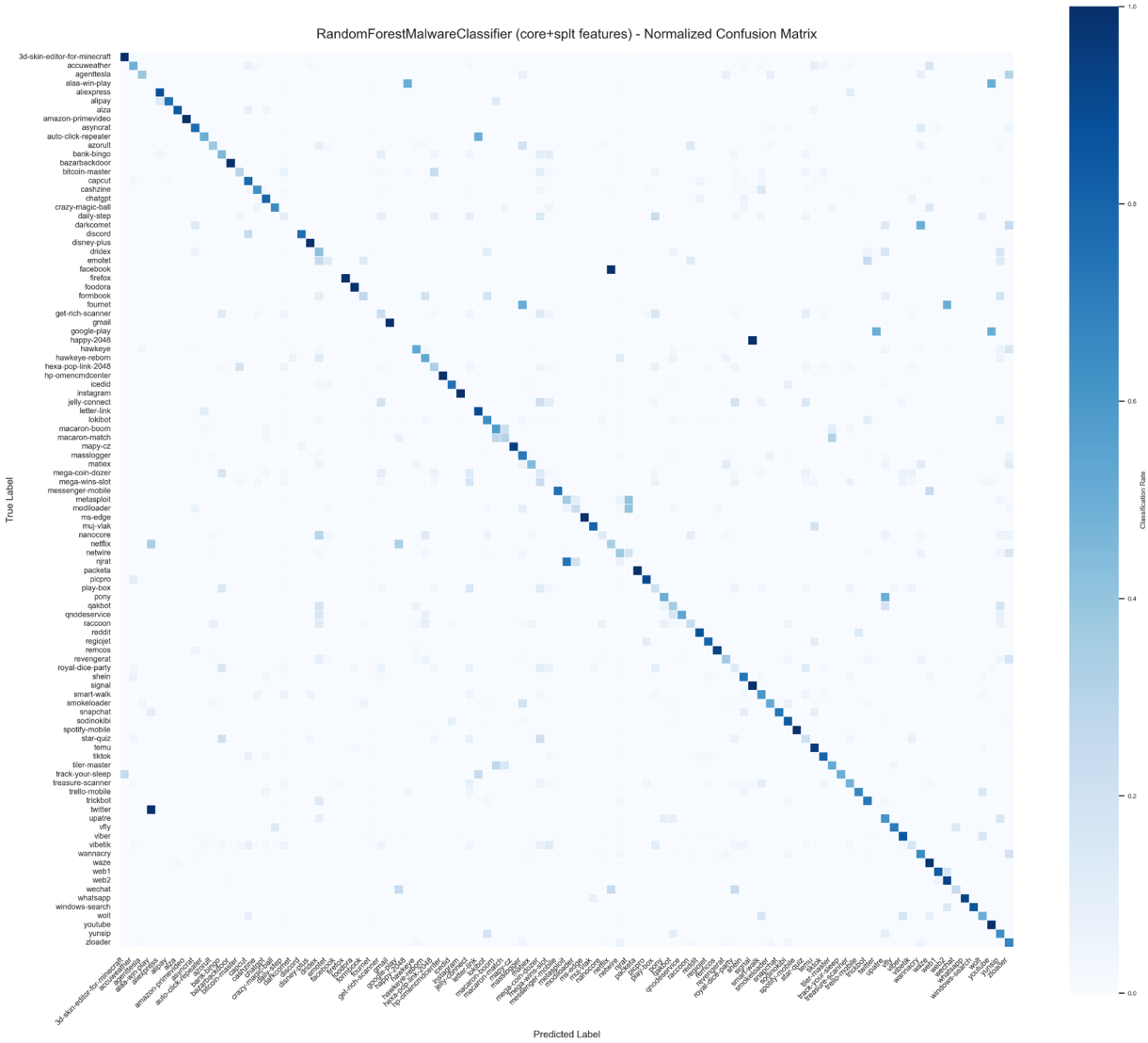


Fig. 2: Normalized confusion matrix for full multiclass classification (Random Forest, combined features). Darker diagonal entries indicate successful classification, while off-diagonal patterns reveal systematic confusions between related families.

TABLE VII: Malware-Only Multiclass Classification Performance (59 Families). Excluding benign classes reduces macro F1 (48.71% vs. 54.81%), reflecting inter-malware similarity and the loss of easily separable benign traffic.

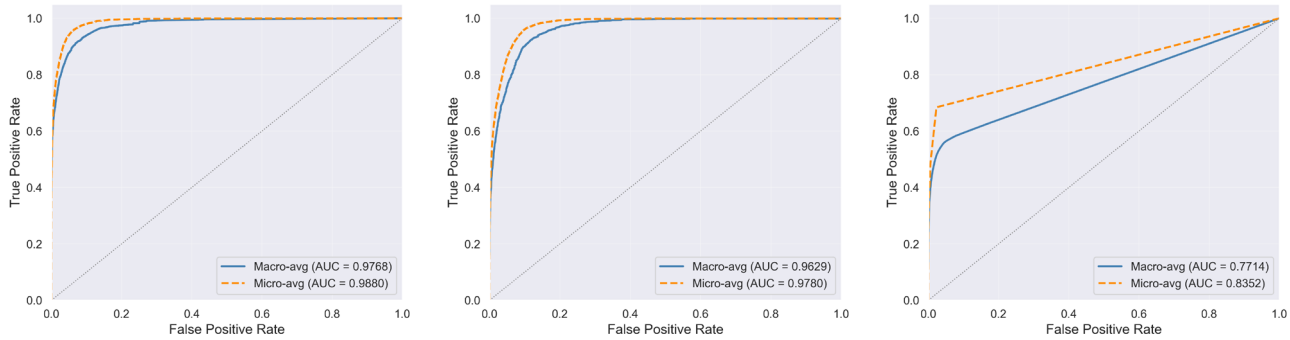
Feature Set	Model	Accuracy	Precision	Recall	F1-Score
Core	NN	0.4052	0.2573	0.2278	0.2121
	RF	0.5886	0.4843	0.4960	0.4764
	FAISS	0.4821	0.3535	0.3571	0.3412
SPL	NN	0.4099	0.2641	0.2243	0.2198
	RF	0.5083	0.3809	0.3944	0.3716
	FAISS	0.3998	0.2736	0.2495	0.2494
Combined	NN	0.4384	0.3288	0.2567	0.2621
	RF	**0.5998**	**0.4958**	**0.5126**	**0.4871**
	FAISS	0.4481	0.3074	0.3008	0.2916

Malware-only macro F1 is lower than full multiclass (48.71% vs. 54.81%).

Fig. 4 shows the normalized confusion matrix for malware-only classification, revealing the patterns of inter-malware confusion across 59 families.

Per-class analysis of the same model shows that perfect accuracies are concentrated in families with very small test support, while several moderate-support families exhibit systematic confusion (e.g., *mega-wins-slot*: 31 test samples, 0% accuracy; *jelly-connect*: 27, 11.1%; *njrat*: 15, 0%). In contrast, well-represented families such as *sodinokibi* (369 test samples) exceed 82% accuracy.

The counterintuitive result—that malware-only classification performs worse than full multiclass—presents several notable patterns. Malware-only classification achieves 48.71% macro F1 compared to 54.81% in full multiclass, representing



(a) Random Forest (b) Neural Network (c) FAISS k-NN

Fig. 3: ROC curves for multiclass classification (101 classes) using combined features. Macro-averaged (solid) and micro-averaged (dashed) curves show Random Forest maintains strong discriminability despite lower F1-scores.

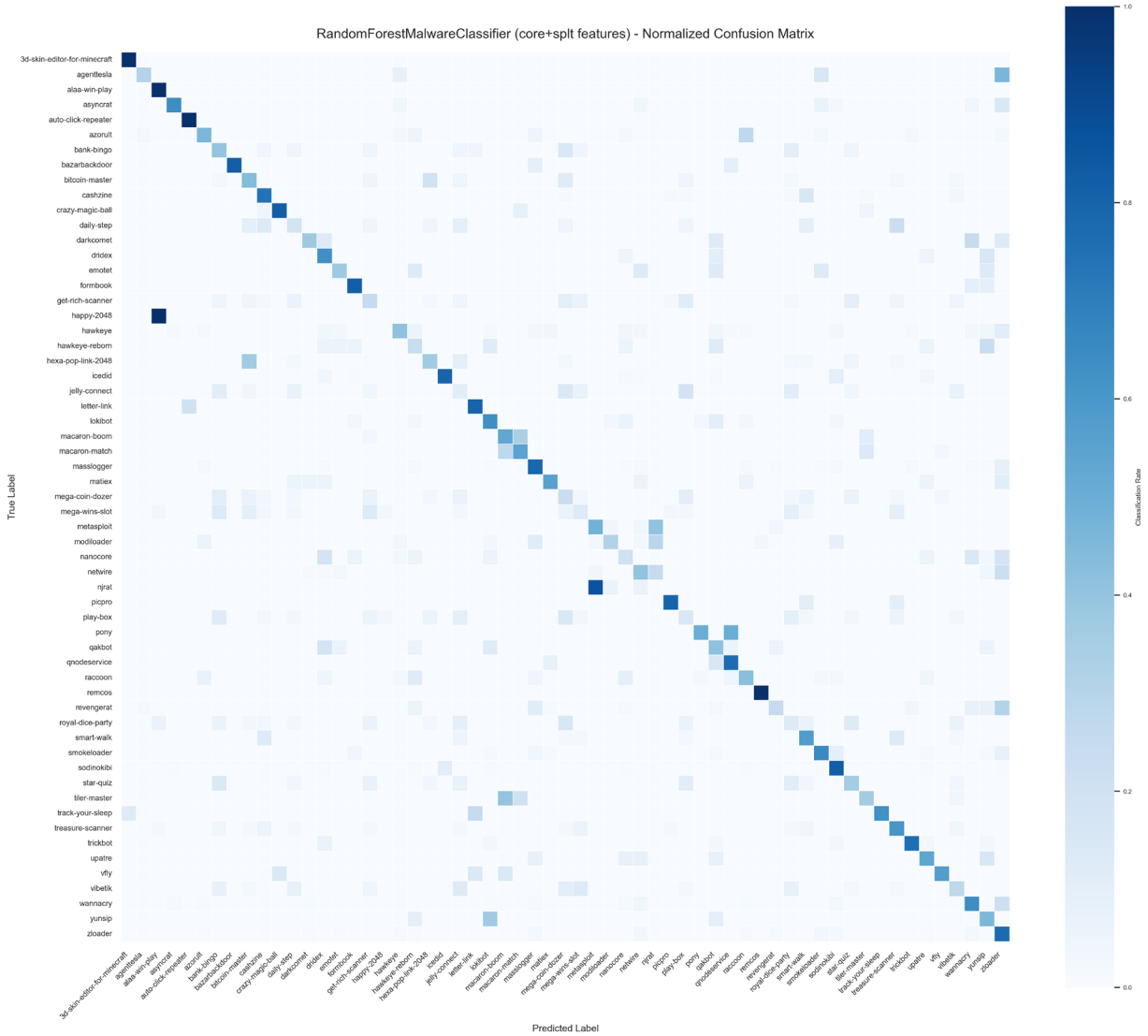
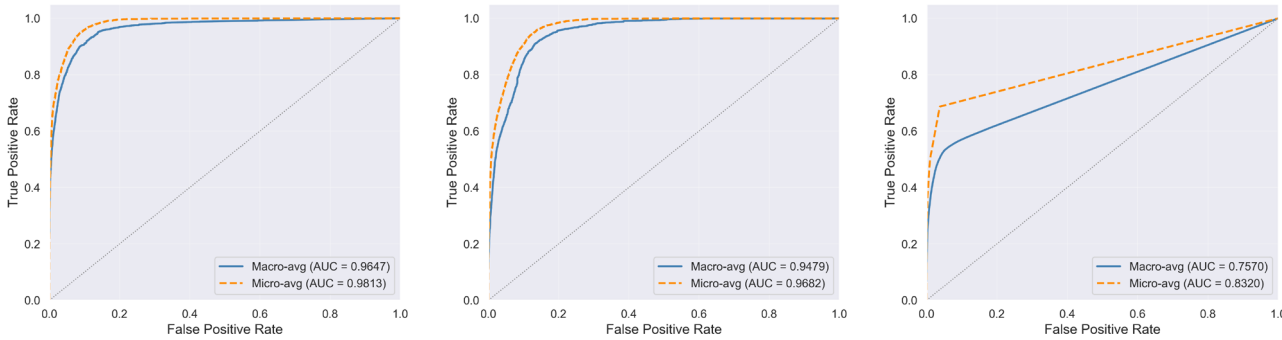


Fig. 4: Normalized confusion matrix for malware-only multiclass classification (Random Forest, combined features) across 59 malware families.



(a) Random Forest (b) Neural Network (c) FAISS k-NN

Fig. 5: ROC curves for malware-only classification (59 classes) using combined features. Macro-averaged (solid) and micro-averaged (dashed) curves show Random Forest maintains strong discriminability for distinguishing between malware families.

a 6.1 percentage point disadvantage. Core flow statistics significantly outperform SPL features in the malware-only context (47.64% vs. 37.16% macro F1 with Random Forest; +10.48 percentage points), a larger gap than observed in full multiclass (+3.12 pp).

Model degradation from binary to malware-only classification is substantial across all approaches: Random Forest drops from 98.11% to 48.71% macro F1 (−49.40 pp), FAISS from 93.14% to 29.16% (−64.00 pp), and Neural Networks from 93.58% to 26.21% (−67.47 pp). The combined feature set improves macro F1 by +1.07 percentage points over core features (48.71% vs. 47.64%), a smaller gain than observed in full multiclass (+2.42 pp).

ROC-AUC Analysis: Table VIII presents ROC-AUC scores for malware-only classification, computed using macro-averaged One-vs-Rest scoring across 59 malware families.

TABLE VIII: ROC-AUC Scores for Malware-Only Classification (59 Classes). Random Forest achieves 0.9647 ROC-AUC, indicating strong discriminability despite lower F1-scores.

Model	Core	SPL	Combined
Neural Network	0.9355	0.9389	0.9478
Random Forest	**0.9614**	**0.9430**	**0.9647**
FAISS k-NN	0.8146	0.7376	0.7570

Similar to full multiclass, the high ROC-AUC (0.9647) contrasting with moderate F1-score (0.4871) indicates that misclassifications occur among similar malware families with comparable probability scores rather than confident errors. Fig. 5 visualizes the macro-averaged and micro-averaged ROC curves for all three models.

D. Comparison with TLS Fingerprinting

Table IX contrasts theoretically derived bounds for TLS fingerprinting with empirical flow-based classification results on the same corpus. TLS entries are upper bounds computed from our NFStream-cleaned dataset: precision bound is 1 − overlap with overlap = 0.38%, recall bound equals malware family coverage (64.9%), and F1 bound is their harmonic mean. Flow-based entries are test-set metrics for Random Forest with combined features. The key distinction: TLS fingerprinting requires handshake inspection and abstains when no match exists, while flow-based classification uses only traffic statistics and produces a prediction for every flow. Consequently, per-instance accuracy cannot be computed for fingerprinting without instance-level predictions.

These results highlight that flow-based methods substantially extend coverage and recall, while fingerprinting remains fundamentally constrained by family uniqueness and handshake visibility.

E. Random Forest: Interpretability and Stability

The preceding results establish Random Forest as the consistently best-performing model across all tasks and feature configurations. We therefore focus our interpretability and stability analyses on Random Forest, which additionally provides natural feature importance measures through its ensemble structure.

1) Feature Importance Analysis: To provide interpretability into the classification decisions, we extracted feature importances from the Random Forest models across all three classification tasks. Table X present the top 10 features for each task using combined features, averaged across 5-fold cross-validation.

The feature importance analysis reveals task-dependent patterns. For *binary classification*, packet size features dominate: the top three features (bidirectional_max_ps, dst2src_max_ps, ps_4) each contribute 10.7–10.9% importance, collectively accounting for 32% of total importance. Both aggregate statistics and early sequential packet sizes appear in the top 10.

For *multiclass classification* (101 families), importance is more evenly distributed across features, with duration and timing features gaining prominence alongside packet sizes. The top feature (ps_4) contributes only 3.8% importance compared to 10.9% in binary classification, indicating that fine-grained family discrimination requires a broader feature combination.

For *malware-only classification* (59 families), temporal features dominate: duration and packet inter-arrival time (PIAT)

TABLE IX: TLS Fingerprinting (bounds) vs. Flow-Based Classification (empirical). Flow-based ML achieves 97.22% recall compared to fingerprinting's 64.9% upper bound—a 32+ percentage point improvement in detection coverage.

Characteristic	TLS Fingerprinting (JA4+JA4S+SNI)	Flow-Based ML (RF + Combined)
Fundamental Capabilities		
ECH-Resilient	No	**Less affected (handshake-independent)**
Handshake Inspection[a]	Requires	**Not required**
Requires Training Data	No	Yes
Handles Unknown Patterns	No (no match)	**Yes (prediction)**
Binary Detection Performance		
Precision (estimated)[b]	∼99.62%	99.02%
Recall (maximum)[c]	≤64.9%	**97.22%**
F1-Score	≤78.6%	**98.11%**
Family Attribution (Multiclass, 101 families)		
Malware Coverage	64.9%	**100%**[d]
Benign Coverage	93.1%	**100%**[d]
Accuracy	N/A[e]	**61.62%**

[a] TLS fingerprinting depends on ClientHello/ServerHello fields; flow-based classification does not inspect handshake fields.
[b] Based on 0.38% overlap between malware and benign fingerprints.
[c] Limited by 64.9% family coverage; 35.1% of families lack unique fingerprints.
[d] Can attempt classification for all families, though accuracy varies.
[e] Cannot compute without instance-level predictions.

TABLE X: Top 10 Feature Importances (Random Forest, Combined Features) across all classification tasks. Binary classification is dominated by packet size features (top 3 = 32%), multiclass shows more even distribution with timing features gaining prominence, and malware-only is dominated by temporal features (8 of top 10).

(a) Binary

Rank	Feature	Imp.
1	bidirectional_max_ps	0.109
2	dst2src_max_ps	0.108
3	ps_4	0.107
4	ps_6	0.066
5	ps_7	0.056
6	src2dst_max_ps	0.051
7	src2dst_stddev_ps	0.025
8	ps_8	0.023
9	ps_2	0.023
10	ps_1	0.019

(b) Multiclass (101)

Rank	Feature	Imp.
1	ps_4	0.038
2	src2dst_max_ps	0.034
3	src2dst_stddev_ps	0.029
4	bidirectional_duration_ms	0.029
5	src2dst_duration_ms	0.028
6	bidirectional_max_ps	0.028
7	src2dst_bytes	0.027
8	dst2src_duration_ms	0.027
9	src2dst_max_piat_ms	0.026
10	dst2src_mean_ps	0.026

(c) Malware-Only (59)

Rank	Feature	Imp.
1	src2dst_duration_ms	0.037
2	bidirectional_duration_ms	0.037
3	src2dst_max_ps	0.032
4	bidirectional_max_piat_ms	0.032
5	src2dst_max_piat_ms	0.032
6	dst2src_max_piat_ms	0.032
7	dst2src_duration_ms	0.031
8	dst2src_mean_piat_ms	0.030
9	bidirectional_mean_piat_ms	0.029
10	src2dst_bytes	0.029

features occupy 8 of the top 10 positions. This shift suggests that distinguishing between malware families relies heavily on communication timing patterns—likely reflecting differences in command-and-control polling intervals, data exfiltration rates, and protocol-specific timing behaviors.

Critically, these top features are largely observable without TLS handshake fields, since they capture transport-layer characteristics rather than ClientHello/ServerHello parameters. The task-dependent feature importance patterns explain why combined features consistently outperform single feature sets: binary detection benefits from strong packet size signals, while family attribution requires the complementary timing information.

2) Stability Analysis: To assess result stability and transferability across different data partitions, we performed 5-fold stratified cross-validation for the best-performing model (Random Forest) across all three classification tasks. Table XI presents the cross-validation results, showing consistent performance across folds.

The cross-validation results demonstrate stability across all tasks. For binary classification, F1-score standard deviations are at most 0.0011, indicating highly stable results. The multiclass tasks show higher variance (F1 std 0.011–0.021), reflecting sensitivity to class imbalance and fold composition—this is expected given the long-tailed distribution with 18 classes having fewer than 20 samples. Importantly, the 5-fold mean F1-scores closely match single-split results (e.g., binary combined: 0.9836 CV vs. 0.9811 single-split), confirming the reliability of our primary evaluation protocol.

F. Summary of Experimental Findings

Across all experiments, several consistent patterns emerge. Flow-based classification achieves near-perfect binary detection, with Random Forest models exceeding 98% F1 (98.36% in 5-fold CV) and 99.6% ROC-AUC, while multiclass family attribution remains more challenging, reaching 54.81% F1 across 101 classes and 48.71% across 59 malware families. Cross-validation confirms result stability across all tasks, with binary F1 standard deviations below 0.002 and multiclass standard deviations of 0.011–0.021. Random Forest consistently outperforms neural networks and k-NN, particularly

TABLE XI: 5-Fold Cross-Validation Results (Random Forest). Low standard deviations confirm stable, representative results across all tasks. Binary classification shows F1 std $\leq$0.0011; multiclass tasks show higher variance (std 0.011–0.021) reflecting class imbalance sensitivity.

Task	Features	Accuracy	F1-Score	ROC-AUC
Binary	Core	0.9682±0.0016	0.9796±0.0010	0.9942±0.0007
	SPL	0.9719±0.0016	0.9819±0.0011	0.9961±0.0005
	Combined	**0.9744±0.0011**	**0.9836±0.0007**	**0.9961±0.0002**
Multiclass (101 classes)	Core	0.6080±0.0028	0.5448±0.0111	0.9711±0.0046
	SPL	0.5587±0.0067	0.4939±0.0108	0.9634±0.0038
	Combined	**0.6268±0.0075**	**0.5694±0.0181**	**0.9751±0.0035**
Malware-only (59 classes)	Core	0.5819±0.0106	0.4806±0.0203	0.9687±0.0027
	SPL	0.5014±0.0102	0.3759±0.0164	0.9440±0.0029
	Combined	**0.5911±0.0094**	**0.4888±0.0200**	**0.9694±0.0024**

under class imbalance, confirming the suitability of tree ensembles for tabular flow features. Feature importance analysis reveals task-dependent patterns: packet size statistics dominate binary detection, while temporal features (duration, inter-arrival times) become more important for family attribution—all features remain observable under ECH deployment. Most importantly, flow-based methods substantially extend detection coverage and recall compared to TLS fingerprinting, which remains fundamentally constrained by family uniqueness and handshake visibility. These findings establish flow features as a robust, ECH-resilient foundation for network-based malware detection.

V. DISCUSSION

A. Performance Analysis and Operational Implications

Our results highlight a clear distinction on this corpus: binary malware detection reaches very high performance, while fine-grained family attribution remains challenging. The performance patterns across binary, multiclass, and malware-only tasks (Tables III, V and VII) reveal important insights about feature complementarity, model suitability, and deployment considerations. Binary detection achieves consistently high F1-scores across all models (>92.5%) with Random Forest reaching 98.11% using combined features, indicating that flow-based features are well-suited for malware vs. benign detection. For binary classification, core flow statistics and SPL features achieve nearly identical performance (97.78% vs. 97.84% F1 with Random Forest), with combined features providing minimal improvement (+0.33 percentage points).

This contrasts with multiclass scenarios where core features consistently outperform SPL features—by +3.12 points in full multiclass and +10.48 points in malware-only classification. Combined features yield only modest gains in multiclass settings (+2.42 points and +1.07 points respectively), suggesting limited complementarity for family discrimination beyond core flow statistics.

Neural networks consistently underperform Random Forest across all tasks, with performance gaps widening as task complexity increases: from 4.5–5.2 points in binary classification to 25.92 points in full multiclass (28.89% vs. 54.81%). This is consistent with overfitting risks under high class counts (101), tabular feature regimes (33–58 input features: 33 core + 25 SPL), and pronounced class imbalance (median 91 samples/class; min 5, max 1845; 18 classes with <20 samples, 32 with <50). Tree ensembles benefit from implicit feature selection, robustness via bagging, and non-linear interactions without heavy regularization. Practical mitigations for neural models include class-weighted loss, mild L2/dropout or label smoothing, and careful width/depth tuning for shallow MLPs; we applied early stopping, and leave broader NN ablations to future work.

For operational deployment, model selection depends on performance requirements: Random Forest maintains both high precision (98.66–99.02%) and high recall (96.91–97.22%) for binary detection, while FAISS k-NN offers strong recall (94.64–95.25%) at the cost of lower precision (91.02–93.38%) in settings prioritizing threat detection over false positive minimization. To ensure methodological rigor, we validated JA4/JA4S extraction against the official JA4+ suite (99.98% flow matching, 100% hash conformance) and used a unified filtering/evaluation pipeline with fixed seeds for full reproducibility.

Malware-only family attribution (59 classes) reaches 48.71% macro F1 with Random Forest, lower than full multiclass (54.81%). This counterintuitive result likely stems from: (1) removal of easily separable benign classes reducing macro-F1 averages; (2) inter-malware similarity and long-tail class imbalance causing systematic confusion in moderate-support families (e.g., *mega-wins-slot*: 31 test samples, 0%; *jelly-connect*: 27, 11.1%; *njrat*: 15, 0%), as shown in Figs. 2 and 4. The similarity among malware families reflects shared operational requirements—many malware types exhibit similar communication patterns when sharing command-and-control infrastructure, common development frameworks, or similar operational objectives, making flow-based discrimination inherently challenging.

B. Dataset Suitability and Generalizability

The dataset provided by Matoušek et al. [10] represents one of the most comprehensive publicly available corpora for TLS-based malware detection research, encompassing 101 families across desktop/mobile malware and benign applications. However, several characteristics warrant discussion regarding result generalizability.

Class Distribution. The dataset exhibits a long-tailed distribution typical of real-world malware collections: median sup-

port is 91 samples per family, but 18 families have fewer than 20 samples and 32 have fewer than 50. This imbalance reflects operational reality—some malware families generate abundant traffic while others are rare—but complicates evaluation. We address this through macro-averaged metrics that weight all families equally, ensuring rare families are not masked by high-frequency ones. The cross-validation results (Table XI) demonstrate that performance is stable across different data partitions, with binary F1 standard deviations below 0.002 and multiclass standard deviations of 0.011–0.021 reflecting sensitivity to fold composition under class imbalance.

Temporal Scope. The dataset captures malware behavior at specific collection periods. Malware evolves continuously; new variants may exhibit different communication patterns than those in the training corpus. Our feature importance analysis suggests that the dominant discriminative features—packet size distributions—reflect fundamental protocol-level behaviors that may be more stable than application-specific patterns. However, longitudinal validation on temporally disjoint datasets remains important future work.

Traffic Mix. The corpus contains controlled capture conditions that may not reflect operational network diversity (e.g., NAT traversal, middlebox interference, variable network conditions). Flow-level statistics may exhibit different distributions in high-latency or lossy environments. Deployment in diverse network conditions should be validated empirically.

Generalization to Unseen Families. Our evaluation uses closed-set classification where all test families appear in training. Open-set scenarios—detecting malware from previously unseen families—require different evaluation protocols. The high ROC-AUC scores (0.99+) for binary detection suggest strong separation between malware and benign distributions, which may support open-set generalization, but explicit evaluation is needed.

Despite these considerations, we believe the dataset is appropriate for this study's primary contribution: demonstrating that flow-level features provide handshake-independent detection capability that substantially exceeds TLS fingerprinting bounds on this corpus. The methodological framework—validated extraction pipelines, cross-validation stability assessment, and systematic comparison—establishes reproducible baselines that future work can extend with additional datasets.

C. Future Work and System Enhancements

To improve family-level attribution beyond the current 48.71–54.81% F1-score range, several enhancement strategies merit investigation. Flow features should be augmented with complementary signals such as DNS query patterns, TLS certificate characteristics, and temporal behavioral sequences that capture attack progression over time.

Methodological improvements should address class imbalance through targeted strategies: class-weighted loss functions for neural networks, intelligent resampling techniques, and flow-weighted evaluation metrics that account for traffic volume differences between families. Hybrid detection pipelines combining fast fingerprint lookups with ML-based fallback classification could leverage the strengths of both approaches.

For a directly comparable TLS fingerprinting baseline, one should construct the fingerprint database from the training split only and evaluate deterministic lookup on the test split, reporting coverage (abstention rate) alongside precision/recall and top-k accuracy. Open-set handling (predicting "Unknown" for unseen families) should be included for realistic deployment scenarios.

Future ablation studies should employ rigorous cross-validation to quantify performance variance and assess the stability of family attribution across different temporal periods. A systematic analysis of confusion matrix clusters can guide feature engineering specifically tailored to problematic family pairs that exhibit persistent misclassification patterns. Taken together, these directions suggest that while binary detection is largely solved, advancing reliable multi-family attribution will require richer features, better imbalance handling, and hybrid pipelines.

D. Limitations

Our evaluation reports macro-averaged metrics, which weigh classes equally and can be sensitive to long-tailed class distributions and very small supports. While we supplement the primary 80-20 split results with 5-fold cross-validation to assess stability (showing binary F1 standard deviations below 0.002 and multiclass standard deviations of 0.011–0.021), the cross-validation was performed for Random Forest across all three tasks; broader CV analysis across all models would strengthen generalizability claims. We applied balanced class weighting in Random Forest but did not systematically evaluate other rebalancing strategies (e.g., SMOTE, class-weighted losses for neural networks), leaving potential performance gains unexplored.

We did not evaluate adversarial robustness (e.g., malware authors deliberately shaping traffic to mimic benign patterns), which remains an important open question for deployment. The dataset captures malware at specific temporal snapshots; concept drift over time may degrade performance on newer malware variants. Finally, compute/runtime characteristics are not benchmarked here and are left to future system evaluations.

E. TLS Fingerprinting vs. Flow ML

Table IX contrasts theoretically derived bounds for TLS fingerprinting (precision bound from 0.38% overlap, recall bound from 64.9% family coverage, F1 bound 78.6%) with empirical flow-based classification results (RF + combined features). The key limitation of fingerprinting is structural: roughly one-third of malware families lack unique TLS signatures, so recall cannot exceed 64.9% even under ideal conditions. In contrast, flow ML attains 97.22% recall, reducing missed detections by over 32 percentage points.

ECH resilience is a critical differentiator. As ECH adoption advances in major browsers and CDNs, methods that depend on inspecting ClientHello/ServerHello fields may lose visibility into the very attributes they use. This trend has been highlighted in multiple surveys [4], [5], which point out that fingerprinting approaches are increasingly brittle as TLS evolves. In contrast, flow statistics remain largely observable

without handshake fields, which should help preserve efficacy as privacy technologies evolve. We do not claim fingerprinting is immediately obsolete under ECH, but its dependence on handshake fields creates a structural limitation that flow-based methods avoid.

Adaptability also differs. Fingerprinting is a deterministic lookup that abstains on previously unseen patterns, whereas learned models may generalize to novel variants that retain behavioral characteristics even when TLS signatures change. Similar observations appear in recent NDSS work [18], which shows that unknown malicious traffic can be detected in real time from flow characteristics even without handshake visibility. This flexibility carries trade-offs: ML requires labeled training data, periodic retraining to handle concept drift, and higher computational resources than hash lookups. For an apples-to-apples baseline, we outline in Future Work a train/test split evaluation for fingerprinting that reports coverage (abstention rate) alongside precision/recall and top-k accuracy, including open-set handling.

VI. Conclusion

This paper presented a systematic evaluation of flow-based statistical features as a handshake-independent complement to TLS fingerprinting for malware detection in encrypted traffic. Across 27 configurations, Random Forest models with combined flow statistics and sequential packet lengths achieved 98.11% F1 in binary detection and 54.81% macro F1 in full 101-class family attribution—substantially exceeding the theoretical recall bound of 64.9% imposed by fingerprinting coverage limits. These results indicate that while binary detection reaches very high performance on this corpus, flow-based features also retain meaningful discriminative power for fine-grained attribution and are less dependent on handshake visibility.

Our evaluation emphasized reproducibility and methodological rigor: JA4/JA4S extraction was validated against the official JA4+ suite, fingerprinting bounds were quantified, and all experiments were performed within a unified pipeline with fixed seeds. These contributions provide reliable performance baselines and support flow-level statistics as a practical foundation for network security monitoring as encryption evolves. Future work will explore hybrid pipelines, integration of complementary side-channel signals, and longitudinal studies to strengthen family-level attribution and operational deployment.

Code Availability

The complete source code, experimental configurations, and reproducibility pipelines supporting this work are publicly available at [22].

Dataset Availability

The complete dataset (16,542 flows, 89 features) is publicly available at [22]. The dataset is an enriched reprocessing of the malware traffic captures originally collected by Matoušek et al. [10], extending the original 15 TLS handshake-focused features with 58 flow-level classification features (33 statistical flow metrics and 25 sequential packet length features), plus 24 TCP flag features and metadata columns extracted using NFStream.

A key distinguishing feature of our dataset is the *temporal flexibility* provided by sequential packet-level (SPLT) features, not typically available in flow datasets. While the 33 core flow statistics represent aggregate measurements from complete flows, the SPLT features preserve packet sizes, directions, and inter-arrival times for the first 25 packets in temporal order. This enables reconstruction of partial flows at any cutoff point (k=1 to 25) without accessing original PCAPs. Researchers can simulate early detection scenarios—computing volumetric, temporal, and statistical features from only the first k packets—to study detection accuracy versus latency trade-offs, progressive classification strategies that adapt observation windows based on confidence, and inline blocking feasibility for network security appliances.

This capability supports a growing research direction in real-time malware detection where classification decisions must be made within milliseconds of connection establishment, and distinguishes our dataset from traditional flow collections that provide only aggregate statistics from completed flows.

Acknowledgment

This work has been part of Celtic-Next project RAI-6Green: Robust and AI Native 6G for Green Networks with project-id: C2023/1-9 funded by 2024-1.2.6-EUREKA-2024-00009.

References

[1] FoxIO-LLC, JA4+: A suite of network fingerprinting methods, https://github.com/FoxIO-LLC/ja4, Accessed: 2025-09-05.

[2] G. Gomez *et al.*, "Unsupervised detection and clustering of malicious tls flows," *Security and Communication Networks*, vol. 2023, pp. 1–17, 2023. DOI: 10.1155/2023/3676692.

[3] B. Anderson *et al.*, "Deciphering malware's use of tls (without decryption)," *Journal of Computer Virology and Hacking Techniques*, vol. 14, no. 3, pp. 195–211, 2017. DOI: 10.1007/s11416-017-0306-6.

[4] C. Oh *et al.*, "A survey on tls-encrypted malware network traffic analysis applicable to security operations centers," *Applied Sciences*, vol. 12, no. 1, p. 155, 2021. DOI: 10.3390/app12010155.

[5] Z. Wang *et al.*, "Machine learning for encrypted malicious traffic detection: Approaches, datasets and comparative study," *Computers & Security*, vol. 113, p. 102 542, 2022. DOI: 10.1016/j.cose.2021.102542.

[6] C. Novo and R. Morla, "Flow-based detection and proxy-based evasion of encrypted malware c2 traffic," in *Proceedings of the 13th ACM Workshop on Artificial Intelligence and Security*, ser. CCS '20, 2020, pp. 83–91. DOI: 10.1145/3411508.3421379.

[7] O. Barut *et al.*, "Machine learning based malware detection on encrypted traffic: A comprehensive performance study," in *Proceedings of the 7th International Conference on Networking, Systems and Security (NSysS '20)*, 2020, pp. 45–55. DOI: 10.1145/3428363.3428365.

[8] J. Althouse. "Tls fingerprinting with ja3 and ja3s." Salesforce Engineering blog post.

[9] J. Althouse. "Ja4+ network fingerprinting." Blog post, FoxIO.

[10] P. Matoušek *et al.*, "Experience report: Using ja4+ fingerprints for malware detection in encrypted traffic," in *2024 20th International Conference on Network and Service Management (CNSM)*, 2024, pp. 1–5. DOI: 10.23919/cnsm62983.2024.10814358.

[11] P. Matoušek *et al.*, "On reliability of ja3 hashes for fingerprinting mobile applications," in *Digital Forensics and Cyber Crime*. 2021, pp. 1–22. DOI: 10.1007/978-3-030-68734-2_1.

[12] Y. R. Siwakoti and D. B. Rawat, "Detecting malicious traffic using ja3 fingerprints attributed ml approach," in *2024 IEEE 44th International Conference on Distributed Computing Systems Workshops (ICDCSW)*, 2024, pp. 128–133. DOI: 10.1109/ICDCSW63686.2024.00024.

[13] T. van Ede *et al.*, "Flowprint: Semi-supervised mobile-app fingerprinting on encrypted network traffic," in *Proceedings 2020 Network and Distributed System Security Symposium*, ser. NDSS 2020, 2020. DOI: 10.14722/ndss.2020.24412.

[14] A. Theofanous *et al.*, "Fingerprinting the shadows: Unmasking malicious servers with machine learning-powered tls analysis," in *Proceedings of the ACM Web Conference 2024*, ser. WWW '24, 2024, pp. 1933–1944. DOI: 10.1145/3589334.3645719.

[15] B. Anderson and D. McGrew, "Identifying encrypted malware traffic with contextual flow data," in *Proceedings of the 2016 ACM Workshop on Artificial Intelligence and Security (AISec)*, 2016, pp. 35–46. DOI: 10.1145/2996758.2996768.

[16] M. Piskozub *et al.*, "Malalert: Detecting malware in large-scale network traffic using statistical features," *ACM SIGMETRICS Performance Evaluation Review*, vol. 46, no. 3, pp. 151–154, 2019. DOI: 10.1145/3308897.3308961.

[17] M. Yeo *et al.*, "Flow-based malware detection using convolutional neural network," in *2018 International Conference on Information Networking (ICOIN)*, 2018, pp. 910–913. DOI: 10.1109/ICOIN.2018.8343255.

[18] C. Fu *et al.*, "Detecting unknown encrypted malicious traffic in real time via flow interaction graph analysis," in *Proceedings 2023 Network and Distributed System Security Symposium*, ser. NDSS 2023, 2023. DOI: 10.14722/ndss.2023.23080.

[19] H. Kim et *al.*, "Revisiting tls-encrypted traffic fingerprinting methods for malware family classification," in *2022 International Conference on Information and Communication Technology Convergence (ICTC)*, 2022, pp. 1273–1278. DOI: 10.1109/ICTC55196.2022.9952872.

[20] S. Yu and Y. Won, "A survey of methods for encrypted network traffic fingerprinting," *Mathematical Biosciences and Engineering*, vol. 20, no. 2, pp. 2183–2202, 2022. DOI: 10.3934/mbe.2023101.

[21] Z. Aouini and A. Pekar, "Nfstream: A flexible network data analysis framework," *Computer Networks*, vol. 204, p. 108 719, 2022. DOI: 10.1016/j.comnet.2021.108719.

[22] FlowFrontiers, *ECH-Resilient Malware Detection via Flow-Level Statistical Features - Digital Artifacts*, https://github.com/FlowFrontiers/MalwareDet-JA4vsFlowStats, 2025.

[23] M. Douze *et al.*, "The faiss library," 2024. *arXiv: 2401.08281* [cs. LG].

Márton Pál Lipcsey-Magyar earned his BSc degree from the Department of Networked Systems and Services at the Budapest University of Technology and Economics, Hungary, in 2026. He is currently pursuing his MSc degree in the same department while also serving as a Research Fellow. His research interests include network and service management and the application of machine learning to various networking domains.

Attila Ármin Madarász earned his BSc degree from the Department of Networked Systems and Services at the Budapest University of Technology and Economics, Hungary, in 2026. He is currently pursuing his MSc degree in the same department while also serving as a Research Fellow. His research interests include network and service management and the application of machine learning to various networking domains.

Adrian Pekar is currently a Senior Data Scientist at CUJO AI, where he develops ML-powered solutions for home networks, focusing on attack detection and encrypted traffic analytics. Previously, he held the position of Associate Professor at Budapest University of Technology and Economics, where he continues to teach part-time. His research interests encompass network traffic flow measurement, machine learning for traffic analytics, federated learning for traffic classification, and cybersecurity applications.