Early-Stage IoT Device Identification Using Passive Network Traffic Analysis

Alex Ciechonski*, Fabio Palmese†, Alessandro E. C. Redondi†, Anna Maria Mandalari*
*{alex.ciechonski.22, a.mandalari}@ucl.ac.uk, †{fabio.palmese, alessandroenrico.redondi}@polimi.it
*University College London, †Politecnico di Milano

***Abstract*—The rapid proliferation of Internet of Things (IoT) devices introduces significant security challenges due to limited visibility and weak device-level guarantees. Accurate and timely identification of devices is essential for enforcing network policies and detecting unauthorised hardware, yet existing approaches often rely on long-term traffic observation, payload inspection, or infrastructure-dependent features. In this paper, we investigate whether IoT devices can be reliably identified during the early stages of network attachment using only passive traffic analysis. We propose a lightweight approach based on flow-level features extracted from metadata, avoiding payload inspection and active probing. Through systematic evaluation across multiple observation windows, we show that device-specific signatures emerge within the first few seconds of communication, enabling high-accuracy identification (up to 99%) across 37 IoT devices. Notably, extending the observation window does not consistently improve performance and may slightly degrade accuracy, indicating that the most discriminative behaviour occurs during initial device startup. These findings demonstrate the feasibility of fast, privacy-preserving IoT device identification at the network edge, supporting real-time enforcement, device inventory, and anomaly detection in practical deployments.**

***Index Terms*—IoT, Device identification, Network Traffic Analysis, Machine Learning**

I. Introduction

The exponential growth of the Internet of Things (IoT) has vastly expanded the attack surface of modern networks. Often deployed with minimal security and limited visibility, these devices are frequent targets for compromise. A core challenge for network operators is maintaining an accurate device inventory; without reliable identification, enforcing access control and detecting unauthorised hardware becomes nearly impossible. While prior work has demonstrated high identification accuracy using long-term traffic aggregation or domain-based features, the effectiveness of early-stage, metadata-only identification remains underexplored. In particular, it is unclear whether reliable device identification can be achieved within the first seconds of network attachment, without relying on payload inspection or infrastructure-dependent signals. In this paper, we investigate identifying IoT devices using only the traffic observed during their initial network attachment phase. During this window, devices typically perform deterministic actions, such as DNS resolution and cloud synchronization, that reflect unique vendor implementation choices. We propose a passive, flow-level identification approach that avoids payload inspection, making it suitable for privacy-sensitive environments. Unlike cloud-centric IoT security frameworks that suffer from high latency our approach enables pervasive intelligence by performing flow-level classification directly on the network gateway.

Our results demonstrate that IoT devices leave stable, discriminative signatures within the first few seconds of activity. Interestingly, we find that extending the observation window does not improve performance, suggesting that identity is most distinct during initial boot-up rather than sustained operation. This allows for rapid, data-efficient enforcement decisions in real-world security applications.

The main contributions of this paper are:

- We demonstrate that IoT devices can be reliably identified using only the first few seconds of network traffic, without payload inspection.
- We provide a systematic analysis of observation window duration, showing that early-stage traffic is more informative than longer-term behavior.
- We design a lightweight, flow-based identification pipeline suitable for deployment on resource-constrained edge devices.
- We evaluate the approach on a 37 IoT device dataset, achieving up to 99% accuracy using short observation windows.

The remainder of this paper is structured as follows. Section II reviews related work on IoT device identification. Section III introduces the threat model. Section IV describes the dataset and experimental setup. Section V details the methodology, including feature extraction and classification model design. Section VI reports the experimental results while the implications and limitations of the approach are discussed in Section VII. Finally, Section VIII concludes the paper with concluding remarks and future research directions.

II. Related Work

A. IoT Device Identification

Over the past decade, extensive research has explored IoT device identification through passive network traffic analysis. Prior work primarily employs supervised machine learning models, including Random Forests, Support Vector Machines, and deep neural networks, to classify devices based on statistical traffic characteristics, often achieving high accuracy given sufficiently large labeled datasets [1], [2], [5]–[7], [9], [10]. To reduce labeling requirements, unsupervised and semi-supervised approaches such as clustering-based fingerprinting

ISBN 978-3-903176-82-9

and representation learning using encoder–decoder architectures have also been proposed [3], [19], though these methods typically lack explicit device-level labeling for enforcement purposes. Feature representations vary across studies, ranging from low-level flow statistics (e.g., packet counts, byte volumes, inter-arrival times) to higher-level contextual attributes such as DNS queries, domains, and port information [4], [17], [18]. While prior work demonstrates strong performance using long-term traffic aggregation and complex feature pipelines, identification performance is highly dependent on deployment context, device population, and feature design [7], [11].

B. DNS-based Traffic Analysis

DNS and DNS-over-HTTPS (DoH) traffic provide valuable signals for network security analysis due to their structured and predictable communication patterns. Even when encrypted, observable metadata such as timing, packet sizes, and flow duration can support classification tasks [4]. Prior work has leveraged DNS behavior for anomaly detection and IoT device identification, particularly during device onboarding when DNS resolution is among the first actions performed after network attachment [8], [12]. While many studies aggregate traffic over extended periods to maximise accuracy, recent findings suggest that short observation windows, sometimes as little as one second, can already yield strong identification performance [16]. These results motivate further investigation into early-stage identification using limited traffic observations.

C. Data Privacy and Leakage

Prior research has demonstrated that passive observation of IoT network traffic can reveal sensitive information about devices and user behaviour, even when payloads are encrypted [13]–[15]. Such metadata may expose device presence, type, and usage patterns, highlighting the privacy implications of traffic-based analysis. These findings underscore the dual-use nature of identification techniques: while they support legitimate security and inventory management, they may also enable surveillance if misused. This reinforces the importance of data minimisation and controlled deployment when designing IoT identification systems.

D. Present Study

In contrast to prior work that relies on long-term traffic aggregation, domain-based fingerprinting, or complex multi-stage pipelines, this study focuses on early-stage IoT device identification using short observation windows. We systematically evaluate how identification accuracy varies as a function of traffic collection duration and demonstrate that high accuracy can be achieved using only the earliest observable network behaviour. Our approach deliberately emphasises low-level statistical flow features, such as timing, volume, and directional characteristics, rather than high-level semantic attributes (e.g., domain names or payload inspection), thereby reducing dependence on infrastructure-specific artefacts.

Furthermore, the system is designed around a modular architecture layering principles, enabling device-agnostic feature extraction and facilitating the straightforward integration of new device classes without modification to the core pipeline. By adopting a defensive network-operator perspective and restricting analysis to lightweight flow-level features, our work complements existing identification and privacy research while emphasizing practicality, extensibility, and early enforcement capabilities.

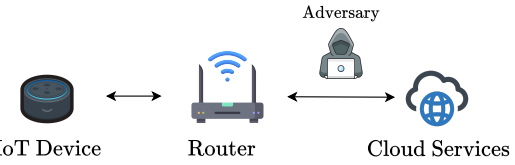

Fig. 1. Threat model overview showing the trusted network operator monitoring IoT device traffic at the gateway.

III. Threat Model

We consider a network environment in which a trusted network operator seeks to identify IoT devices connected to the network using passive traffic observations. The operator has visibility into network metadata at a gateway or monitoring point, such as DNS queries and flow-level statistics, but does not inspect packet payloads or actively interact with devices (as shown in Figure 1).

The objective of the operator is to verify the identity of connected devices in order to support security monitoring, access control, and policy enforcement. We assume that devices are not actively attempting to evade identification and that the network infrastructure itself is trusted.

While similar traffic analysis techniques could potentially be abused by a malicious observer to infer device information, this work focuses on the defensive use of passive identification mechanisms to improve network security and manageability.

IV. Dataset and Experimental Setup

A. Dataset Collection

We collect network traffic from a controlled IoT testbed designed to capture device behaviour immediately following network attachment, as depicted in Figure 2. The testbed consists of multiple IoT devices connected through smart power outlets and a network gateway that passively records all network activity. Device power states are programmatically controlled to enable repeated power-cycling experiments, allowing systematic observation of device initialization behaviour.

Each experiment begins with a device power-on event, followed by continuous traffic capture during the subsequent startup phase. This approach is intended to elicit device-specific communication patterns, such as DNS resolution and initial cloud interactions, which are known to occur shortly after network attachment. Experiments are repeated across multiple days and time periods to capture variability in device

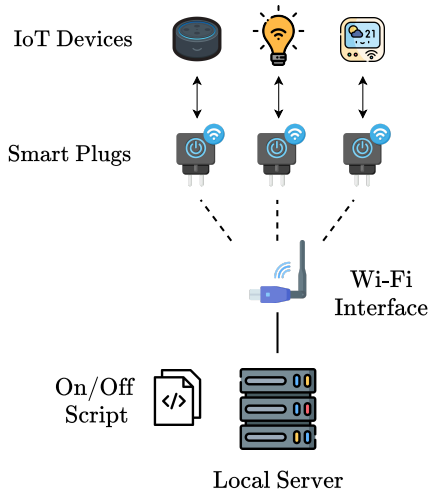

Fig. 2. Experimental testbed showing IoT devices connected through smart power outlets with passive traffic monitoring at the gateway.

behaviour while maintaining consistent experimental conditions.

Our data collection methodology is inspired by prior IoT measurement studies that employ power-cycling and passive traffic monitoring to capture early-stage device communication behaviour [9]. While the overall experimental design follows established best practices, the collected data is used here specifically to study early device identification based on limited observation windows.

B. Experimental Setup

From the collected traffic traces, sessions corresponding to individual device startup events are extracted and processed independently. Each session represents a single device power cycle and includes only traffic observed within a predefined time window following power-on. Flow-level features are computed for each session as described in Section V.

To evaluate the feasibility of early identification, we vary the duration of the observation window and assess classification performance as a function of available traffic. Sessions are split into training and test sets at the session level to prevent information leakage across experiments. All model training and evaluation is performed using these session-level representations.

V. Methodology

A. Feature Justification

Table I lists the 40 flow-level features extracted per flow. Features span five broad categories: volume (packet/byte counts and rates), timing (inter-arrival time statistics), TCP flag counts, payload characteristics (entropy and non-zero byte fraction), and categorical metadata (protocol, port bucket, destination type). Payload entropy is computed over the first 512 bytes to capture encryption and encoding patterns without requiring full payload inspection. Port numbers are bucketed into well-known categories rather than used raw, to improve generalization across deployments.

B. Feature Validation

In the next steps of building the model, we have identified features to be removed. The decisions have been made by checking if features are part of the following criteria: redundant linear combinations, highly correlated features, low variance features and overlapping features.

Redundant linear combinations. The only features that were removed due to them being linear combinations were packets total and bytes total, as they are the combinations of `packets_fwd` and `packets_bwd`, and `bytes_fwd` and `bytes_bwd`, respectively.

Highly correlated features. Linear correlation coefficients have been computed for each pair of features, and those that had a correlation coefficient of at least 0.9 with any other feature have been dropped. Practically speaking, in such analysis, the removed features were the forwrd and backward minimum and maximum values of packet length and inter-arrival times. The practical understanding could be such that the means and standard deviation presented enough information, and the addition of maximal and minimal values added no valuable insights.

Low Variance Features. After testing the variances in the values of features, the removed features were the TCP flags.

Overlapping Derived Features. This is a category of features that can be derived from remaining features; however, they do not exhibit strong linear correlation with any other features. The only feature that has been removed for that reason was `down_up_byte_ratio` since it is the inverse of the `down_up_pkt_ratio`.

C. Data Preprocessing

After the selection of only the relevant features, the next steps involve cleaning the data in the form of removing null values. Then, the data is split between the sessions to prevent occurrences of accuracy degradation due to data leakage—namely, when data coming from a particular session exists both in the training and testing datasets, overfitting is very likely to occur. After doing so the data is split and scaled. Finally, the testing data is being balanced by oversampling.

D. Model

Instead of a single multiclass classifier, we adopt a one-vs-rest architecture composed of binary Random Forest classifiers, where each model is trained to distinguish a specific device from all others. This design allows incremental addition of new devices without retraining the entire model, which is particularly desirable in dynamic IoT environments where new devices are continuously introduced. However, this approach introduces a gradual imbalance in the training data as the number of devices increases, which we discuss further in Section VII. While a multiclass formulation would provide a more compact representation, the one-vs-rest design offers

TABLE I

FLOW-LEVEL FEATURES EXTRACTED PER FLOW AND THEIR VALIDATION STATUS. REMOVAL REASONS: **L** = LINEAR COMBINATION OF OTHER FEATURES; **C** = HIGH PAIRWISE CORRELATION ($r \geq 0.9$); **V** = LOW VARIANCE; **D** = OVERLAPPING DERIVED FEATURE.

Feature	Meaning	Units	Removed
dur	Flow duration	s	—
pkts_fwd, pkts_bwd	Packet counts per direction	count	—
pkts_tot	Total packet count	count	L
bytes_fwd, bytes_bwd	Byte counts per direction	bytes	—
bytes_tot	Total byte count	bytes	L
pktlen_fwd_mean, pktlen_fwd_std	Forward packet length mean & std	bytes	—
pktlen_fwd_min, pktlen_fwd_max	Forward packet length min & max	bytes	C
pktlen_bwd_mean, pktlen_bwd_std	Backward packet length mean & std	bytes	—
pktlen_bwd_min, pktlen_bwd_max	Backwardd packet length min & max	bytes	C
iat_fwd_mean, iat_fwd_std	Forward inter-arrival time mean & std	s	—
iat_fwd_min, iat_fwd_max	Forward inter-arrival time min & max	s	C
iat_bwd_mean, iat_bwd_std	Backward inter-arrival time mean & std	s	—
iat_bwd_min, iat_bwd_max	Backward inter-arrival time min & max	s	C
iat_tot_mean, iat_tot_std	Total inter-arrival time mean & std	s	—
pps, bps	Packet & byte rate	pkt/s, B/s	—
down_up_pkt_ratio	Down/up packet ratio	—	—
down_up_byte_ratio	Down/up byte ratio	—	D
syn_cnt, ack_cnt, fin_cnt, rst_cnt	TCP flags†	count	V
psh_cnt, urg_cnt, ece_cnt, cwr_cnt	TCP flags†	count	V
payload_entropy_fwd, payload_entropy_bwd	Payload Shannon entropy per direction	bits/byte	—
payload_nonzero_frac_fwd, payload_nonzero_frac_bwd	Non-zero payload fraction per direction	$[0, 1]$	—
has_fwd, has_bwd	Traffic presence indicators	binary	—
proto	Transport-layer protocol	categ.	—
sport_bucket, dport_bucket	Port category	categ.	—
internal_dst	Destination is RFC1918	binary	—

†SYN=Synchronise, ACK=Acknowledgement, FIN=Finish, RST=Reset, PSH=Push, URG=Urgent, ECE=ECN-Echo, CWR=Congestion Window Reduced

greater flexibility and supports modular deployment at the network edge, allowing for on-device inference in consumer-grade Wi-Fi routers

Model Architecture. We employ a Random Forest classifier as the primary learning model due to its robustness to feature scaling, ability to model non-linear decision boundaries, and strong performance on tabular network traffic data. Random Forests have been widely used in network security applications and provide a favorable trade-off between interpretability and classification accuracy.

Unless otherwise stated, the base model configuration consists of 100 decision trees, with no explicit limit on tree depth, allowing individual estimators to grow until pure leaf nodes or minimum sample constraints are reached. This configuration enables the model to capture complex interactions between traffic features while relying on ensemble averaging to mitigate overfitting. A fixed random seed is used to ensure reproducibility of all experiments.

Hyperparameters. Hyperparameter optimization is conducted using randomised search with cross-validation on the training set. Rather than exhaustively searching the hyperparameter space, we adopt randomised sampling to efficiently explore a broad range of configurations while limiting computational overhead. The following parameters are tuned: number of trees, maximum tree depth, minimum samples required for node splitting, minimum samples per leaf, and the number of features considered at each split. Parameter ranges are selected based on common best practices in the literature and preliminary experiments.

The final model configuration is selected based on mean cross-validation accuracy and subsequently evaluated on a held-out test set that remains unseen during training and tuning. This evaluation protocol ensures that reported results reflect generalization performance rather than optimization artefacts.

E. Data Storage and Deployment

Since this problem requires that on some occasions the model should be trained on a separate machine to which it is deployed, whereas other times it has to be trained on the same device it is deployed, a cache for intermediate session data has been implemented to speed up the training process in the instances where a different device is used for training to the one the model is deployed to. Namely, the preprocessed data is written to individual parquet files, where each file contains data in one session until a specific number of seconds has passed since the session begun. Also, the sessions metadata is stored in a local key-value store. This allows for data to be quickly retrieved and provide additional insight into the value of different timeframes of collection.

VI. RESULTS

A. Overall Performance

The proposed model achieves a test accuracy of approximately 97%, indicating strong generalisation performance across unseen sessions. Training accuracy approaches 100%,

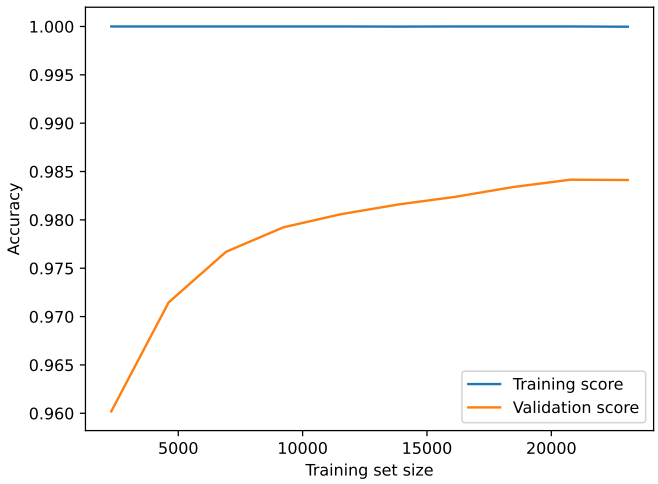

Fig. 3. Learning curve with train/test accuracy at different training set sizes.

reflecting high separability of device-specific traffic patterns in the selected feature space. The relatively small gap between training and test accuracy suggests limited overfitting.

Figure 3 illustrates the learning curve of the Random Forest classifier as a function of training set size. The model reaches high accuracy with a relatively small amount of training data, demonstrating that the extracted features are highly informative for device identification. As the training set size increases, test accuracy exhibits a slight downward trend, reflecting increased behavioural variability across sessions rather than degradation in model capacity. Importantly, the gap between training and test accuracy remains stable, indicating controlled generalisation behaviour.

The observed learning behaviour suggests that early-stage network traffic contains consistent, device-specific signatures that can be learned efficiently. The absence of significant performance gains with additional training data implies that the selected feature set captures the dominant discriminative characteristics of the devices under study. This property is particularly desirable for practical deployments, where labeled data may be limited.

Table II presents detailed classification performance metrics for each device in the testbed. The majority of devices achieve perfect or near-perfect test accuracy, with only a small number exhibiting classification errors. These results demonstrate the effectiveness of the feature set and model architecture for discriminating between diverse IoT devices.

Selected confusion matrices for representative devices are shown in Table III. Perfect classification is observed for the majority of devices, while occasional misclassifications occur primarily between devices from the same vendor family or with similar cloud service dependencies.

B. Minimum Collection Window

To evaluate the minimum observation time required for accurate identification, we trained and evaluated the model using incremental traffic windows ranging from 10 to 105 seconds.

TABLE II
SUMMARY STATISTICS OF BINARY CLASSIFICATION PERFORMANCE ACROSS 37 IOT DEVICE MODELS (TRAIN ACCURACY = 1.000 FOR ALL DEVICES)

Metric	Mean	Std	Min	Max
Test Accuracy	0.987	0.029	0.860	1.000
Precision	0.988	0.033	0.840	1.000
Recall	0.987	0.028	0.900	1.000
F1-Score	0.987	0.029	0.850	1.000
Support	315	226	14	913

TABLE III
ERROR ANALYSIS ACROSS ALL 37 IOT DEVICE CLASSIFIERS

Error Type	Min	Max	Mean	Devices with 0 errors
False Positives	0	17	0.41	29/37 (78.4%)
False Negatives	0	6	0.19	32/37 (86.5%)
Total Errors	0	17	0.59	20/37 (54.1%)

Across all evaluated intervals, identification accuracy remains consistently high, ranging between 98% and 99%. Notably, extending the observation window beyond the initial seconds does not yield consistent performance improvements across devices, and the full window exhibits slightly lower accuracy. These results indicate that the most discriminative device-specific traffic patterns occur shortly after network attachment. Early communication phases, such as DNS resolution and initial cloud interactions, appear sufficient for reliable identification. Additional traffic collected at later stages introduces behavioural variability without providing new identity-relevant information, leading to a marginal reduction in accuracy.

Table IV provides summary statistics. The 30-second window achieves the highest average accuracy (99.4%) with the greatest proportion of devices achieving perfect classification (75.7%).

C. Error Analysis

Misclassifications are primarily observed between devices from the same vendor family or with similar backend service dependencies, suggesting that shared infrastructure can partially obscure device-specific signatures. Nevertheless, these errors remain infrequent and do not significantly impact overall identification accuracy.

VII. DISCUSSION

A. Implications of Early IoT Identification

The results demonstrate that IoT devices exhibit stable and distinctive behaviour immediately after network attachment. High identification accuracy is achievable within short observation windows, indicating that early-stage communication—such as DNS resolution, authentication, and cloud synchronisation—contains strong device-specific signatures. Extending the observation window does not improve performance.

TABLE IV
SUMMARY STATISTICS OF TEST ACCURACY ACROSS TIME WINDOWS

Time Window	Mean Acc.	Std. Dev.	Perfect (100%)
10 seconds	98.7%	2.2%	20/37 (54.1%)
20 seconds	98.2%	3.7%	21/37 (56.8%)
30 seconds	**99.4%**	**1.5%**	**28/37 (75.7%)**
45 seconds	98.9%	1.6%	25/37 (67.6%)
60 seconds	98.4%	2.7%	23/37 (62.2%)
75 seconds	99.2%	1.7%	26/37 (70.3%)
90 seconds	98.9%	1.9%	25/37 (67.6%)
105 seconds	98.9%	1.9%	25/37 (67.6%)
Overall	**98.8%**	**2.2%**	–

The learning behaviour further indicates that these signatures are consistent across sessions and can be captured with relatively modest training data. From a deployment perspective, this supports rapid enforcement decisions at network gateways without prolonged monitoring or intrusive inspection techniques. Early-stage identification is therefore not only sufficient but preferable for practical security applications.

B. Misclassification Patterns and Limitations

Misclassifications occur primarily between devices from the same vendor or those sharing backend infrastructure, highlighting the influence of common cloud services on traffic behaviour. While infrequent, such cases suggest an inherent limitation of purely traffic-based identification when device implementations are closely related.

The evaluation is conducted in a controlled environment with a finite device set and firmware versions. Behavioural changes due to software updates, infrastructure modifications, or adversarial traffic obfuscation may affect long-term performance. Future work should therefore evaluate robustness across more diverse deployments and consider resilience against evasion techniques.

VIII. CONCLUSION

This paper demonstrates that IoT devices can be accurately identified using passive observation of early-stage network traffic. Through systematic evaluation across multiple time windows, we show that device-specific signatures emerge within the first seconds of communication, enabling rapid and data-efficient identification. Our results indicate that extending observation windows provides limited additional benefit, suggesting that early-stage behavior captures the most discriminative characteristics of device identity. By relying solely on flow-level metadata, the proposed approach supports privacy-preserving deployment at network gateways without requiring payload inspection or active probing. These findings highlight the potential of edge-based intelligence for real-time IoT security applications, including device inventory, access control, and anomaly detection.

Future work will focus on extending the evaluation to a larger and more diverse device population, including multiple firmware versions and deployment environments. Additional research is needed to assess robustness against encrypted DNS, traffic shaping, and intentional evasion strategies. Integrating the proposed approach with complementary identification signals may further improve resilience in adversarial settings.

REFERENCES

[1] J. Kotak and Y. Elovici, "IoT device identification using deep learning," in *Computational Intelligence in Security for Information Systems*, Springer, 2019, pp. 76–86.

[2] J. Kotak and Y. Elovici, "IoT device identification based on network communication analysis using deep learning," *Journal of Ambient Intelligence and Humanized Computing*, vol. 14, pp. 1–17, 2022

[3] C. Koball, B. P. Rimal, Y. Wang, T. Salmen, and C. Ford, "IoT device identification using unsupervised machine learning," *Information*, vol. 14, no. 6, p. 320, Jun. 2023, doi: 10.3390/info14060320.

[4] A. Alrawi, C. Lever, M. Antonakakis, and F. Monrose, "IoTFinder: Efficient IoT device identification using network traffic," in *Proc. IEEE European Symposium on Security and Privacy (EuroS&P)*, Genoa, Italy, 2020, pp. 474–489.

[5] M. Mainuddin et al., "IoT device identification based on network traffic characteristics," in *Proc. IEEE Global Communications Conference (GLOBECOM)*, Rio de Janeiro, Brazil, 2022, pp. 6067–6072

[6] R. Kolcun, D. A. Popescu, V. Safronov, P. Yadav, A. M. Mandalari, R. Mortier, and H. Haddadi, "Revisiting IoT device identification," arXiv preprint arXiv:2107.07818, 2021.

[7] K. Kostas, M. Just, and M. A. Lones, "IoTDevID: A behavior-based device identification method for the IoT," *IEEE Internet of Things Journal*, vol. 9, no. 23, pp. 23741–23749, Dec. 2022, doi: 10.1109/JIOT.2022.3191951.

[8] C. Zhang, X. Hu, X. Pan, G. Cheng, R. Li, and H. Wu, "Accurate and early detection of IoT malware via DNS traffic analysis with deep learning," in *Proc. IEEE International Conference on Communications (ICC)*, Montreal, Canada, 2025, pp. 2665–2670, doi: 10.1109/ICC52391.2025.11161323.

[9] S. Pélissier, G. Anselmi, A. K. Mishra, A. M. Mandalari, and M. Cunche, "Enhancing IoT privacy: Why DNS-over-HTTPS alone falls short?" in *Proc. IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, Sanya, China, 2024, pp. 1353–1360, doi: 10.1109/TrustCom63139.2024.00189.

[10] F. Palmese, A. E. C. Redondi, and M. Cesana., "Designing a Forensic-ready Wi-Fi Access Point for the Internet of Things.," IEEE Internet of Things Journal 10.23 (2023): 20686-20702

[11] K. Kostas and R. Y. Kostas, "IoT device identification with machine learning: Common pitfalls and best practices," arXiv preprint arXiv:2601.20548, 2026.

[12] M. Miettinen et al., "IoT sentinel: Automated device-type identification for security enforcement in IoT," in *Proc. IEEE International Conference on Distributed Computing Systems (ICDCS)*, Atlanta, GA, USA, 2017, pp. 2177–2184.

[13] N. Apthorpe et al., "A smart home is no castle: Privacy vulnerabilities of encrypted IoT traffic," arXiv preprint arXiv:1705.06805, 2017.

[14] N. Apthorpe, D. Reisman, S. Sundaresan, A. Narayanan, and N. Feamster, "Spying on the smart home: Privacy attacks and defenses on encrypted IoT traffic," arXiv preprint arXiv:1708.05044, 2017.

[15] J. Ren, et al., "Information exposure from consumer IoT devices: A multidimensional, network-informed measurement approach," in *Proc. ACM Internet Measurement Conference (IMC)*, Amsterdam, Netherlands, 2019, pp. 267–279.

[16] A. Pinheiro, J. Bezerra, C. Burgardt, and D. Campelo, "Identifying IoT devices and events based on packet length from encrypted traffic," *Computer Communications*, vol. 144, pp. 8–17, May 2019.

[17] O. Thompson et al., "Rapid IoT Device Identification at the Edge," in *Proceedings of the 2nd International Workshop on Distributed Machine Learning (DistributedML '21)*, Virtual Event, Germany, Dec. 2021, pp. 1–7. doi: 10.1145/3488659.3493777.

[18] A. Sivanathan et al., "Classifying IoT Devices in Smart Environments Using Network Traffic Characteristics," *IEEE Transactions on Mobile Computing*, vol. 18, no. 8, pp. 1745–1759, Aug. 2019. doi: 10.1109/TMC.2018.2866249.

[19] A. Sivanathan et al., "Generalizable IoT Traffic Representations for Cross-Network Device Identification," arXiv preprint, 2026.