EVADE: A Lightweight Unsupervised Malicious Detection over Encrypted Traffic

David Haynes[†], Phu H. Phung[§], Boyang Wang[†]
[†]University of Cincinnati, [§]University of Dayton
haynesd@mail.uc.edu, phu@udayton.edu, boyang.wang@uc.edu

***Abstract*—Detecting malicious communication in modern networks has become increasingly challenging due to the widespread adoption of encryption protocols, where attackers can hide malicious behaviors. Existing detection methods on encrypted network traffic are often supervised-based and require labeled malicious traffic, which is difficult to obtain. This paper proposes a lightweight, unsupervised method to detect malicious activities over encrypted network traffic. We leverage Principal Component Analysis to reduce the dimensionality of data and utilize Elliptic Envelope to detect anomalies without needing labeled malicious encrypted traffic data. We demonstrate the efficiency and effectiveness of our method through empirical evaluations performed on a large-scale real-world dataset. Specifically, our method shows high precision (98.6%) and high recall (97.6%), surpassing the performance of leading supervised ones. Moreover, our lightweight approach allows it to run 15 times faster in detection than other unsupervised methods, confirming low latency and memory usage even on resource-constrained devices such as Raspberry Pi.**

I. Introduction

Detecting malicious communication in modern networks has become increasingly challenging due to the widespread adoption of encryption protocols such as Transport Layer Security (TLS). Encryption allows attackers to conceal malicious behavior within seemingly benign traffic, rendering traditional deep packet inspection ineffective. As a result, rule-based intrusion detection systems (IDS) that rely on static signatures or handcrafted features often fail to adapt to modern obfuscation techniques and evolving malware communication patterns [1].

To address these limitations, recent research has explored using machine learning models to detect encrypted malicious traffic. In particular, approaches based on supervised neural networks have gained popularity due to their ability to model complex traffic patterns [2]. Despite their performance, these neural network models are often computationally heavy, opaque, and ill-suited for real-time or edge deployment. They struggle to adapt to unseen traffic due to distribution shifts between training and deployment network environments, a fundamental limitation in supervised learning. While some studies, e.g., Rosetta et al. [1], attempted to improve generalization via pre-training or augmentation, they still require careful tuning and large amounts of training data.

In contrast, unsupervised learning approaches avoid these pitfalls by identifying statistical deviations from learned benign behavior without labeling attack network traffic samples [3]. However, many of these methods remain complex or resource-intensive, making them impractical for running detection on IoT or edge devices.

In this paper, we address the technical challenge of detecting malicious communication in encrypted network traffic by developing EVADE, an Elliptic Envelope and PCA-based Anomaly Detection Engine that balances accuracy, adaptability, and efficiency for deployment in resource-constrained environments. Our method is an unsupervised approach combining Principal Component Analysis (PCA) [4] for dimensionality reduction and the Elliptical Envelope algorithm [5] for anomaly detection. PCA enhances computational efficiency by projecting high-dimensional feature spaces into a compact subspace [6]. At the same time, Elliptic Envelope estimates the distribution of benign traffic [7] and uses the Mahalanobis distance [8] to flag anomalous deviations without relying on labeled data. Our contributions are summarized as follows:

- We present a novel method combining PCA and Elliptic Envelope for detecting malware on encrypted network traffic, optimized for efficiency and efficacy.
- We demonstrate that our proposed method outperforms recent supervised models using a real-world data set containing more than 12.8 GB of network traffic generated by 105 physical IoT devices deployed in a smart home environment [9], achieving a higher accuracy of 98.6% compared to their 96.8%. In addition, our method runs over 15 times faster than competing unsupervised methods (2 ms vs. 31 ms).
- We demonstrate the practical deployment of our method on embedded systems, including Raspberry Pi, confirming low latency and memory usage for real-time endpoint security applications.

Reproducibility. Our implementation can be found in `https://github.com/haynesd/evade`.

Paper Organization. The remainder of this paper introduces the system and threat model in Sec. II, reviews related work in Sec. III, and provides background in Sec. IV. Sec. V details our proposed method, Sec. VI evaluates it through experiments and comparisons, and Sec. VII

discusses limitations. We conclude with a summary in Sec. VIII.

II. System and Threat Model

We consider a network where all traffic is encrypted using standard protocols such as TLS. A recent industry study shows that over 90% of web traffic is encrypted, and more than 70% of malware leverages TLS for command-and-control (C2) communications [10]. The proposed detection method aims to identify encrypted traffic generated by malware and operates in two phases: training and detection.

In the training phase, we assume that all traffic is benign, reflecting common practice in anomaly detection. During the detection phase, we assume that benign traffic dominates while malware represents a minority. Public datasets [11], [12], [13] show that malicious flows account for less than 10–15% of total traffic, and industry reports [14], [15] confirm that malicious activity constitutes only a small fraction of overall flows.

We assume that malware has already been installed on a device and is sending encrypted communications to a remote Command and Control (C2) server, operated by either an individual adversary or a nation-state group. Because our proposed detection method cannot decrypt packet payloads, its objective is to differentiate encrypted packets generated by malware from those produced by benign applications. Threat intelligence reports consistently show that modern malware families (e.g., Zeus, TrickBot, Cobalt Strike, APT campaigns) use application-layer encryption such as HTTPS/TLS to conceal C2 traffic within normal encrypted flows. Attacks relying on network-layer encryption (e.g., IPsec) are beyond the scope of this paper, as they require different detection strategies. Thus, assuming encrypted C2 channels reflects the prevailing adversary tradecraft observed in operational environments. [15], [14], [16].

In other words, given a network packet with encrypted payload, the detection method must rely on unencrypted metadata—such as headers, protocol type, packet size, and timing—to perform effective detection.

III. Related Work

The rapid growth of cyber threats targeting encrypted traffic, particularly in IoT networks, has led to significant advancements in malware detection. For example, in [17], Okur and Dener explore classifying normal and malicious traffic in IoT networks using classical supervised learning. Recent studies [2], [18], [19], [20], [21] increasingly use neural networks for malware detection with deep learning to identify malicious activities in network traffic, binaries, and system behaviors. In particular, Choudhury et al. [2] proposed a supervised approach using convolutional neural networks (CNNs) to detect encrypted malicious traffic. Their model extracted spatial features from the packet metadata and achieved high accuracy in the CIC IoT 2022 dataset [22], a cutting-edge dataset designed for profiling, behavior analysis, and vulnerability assessment of various IoT devices operating over diverse communication protocols, including IEEE 802.11 (Wi-Fi), Zigbee, and Z-Wave Liu and Cao [18] utilized a recurrent neural network (RNN) with a feature selection layer to classify encrypted network traffic. Their approach improved recall and robustness to obfuscated attack vectors on the CIC-IDS 2017 dataset, which contains the most up-to-date benign common attacks, similar to real-world data (PCAP) [11] Sun et al. [19] introduced GNN-IDS, a graph neural network–based intrusion detection system that models relationships between packets and devices. By leveraging topological patterns in encrypted network flows, the system detects intrusions with high performance in large-scale network environments.

Deldar and Abadi [20] surveyed deep learning approaches for zero-day malware detection, emphasizing the importance of AutoEncoders, GANs, and ensemble models in generalizing to novel threats. King and Huang [21] proposed Euler, a temporal link prediction model that detects lateral movement across network sessions. By capturing time-sensitive relational patterns, it supports early detection of evasive malware over encrypted traffic. Liu and Cao [18] examined using LSTM networks and neural language models to classify HTTPS-encrypted traffic without relying on payload inspection. These models use time series characteristics and TLS handshake metadata to distinguish between benign and malicious sessions. A recent framework, named Rosetta, leverages TCP-aware augmentation and self-supervised pretraining to enhance encrypted traffic classification, especially in scenarios with limited labeled data and variable session behavior [1].

Despite the promising results of these supervised neural network–based methods, several deficiencies limit their long-term effectiveness in real-world cybersecurity scenarios. First, these models rely heavily on large volumes of labeled training data, often failing to represent emerging or zero-day attack patterns. Second, neural networks frequently suffer from overfitting to specific datasets or attack types, making them brittle when exposed to adversarially crafted inputs or traffic from different environments. Third, their black-box nature complicates interpretability and hinders trust in high-assurance settings where transparent decision-making is critical. Finally, the computational overhead of deep learning architectures can impede deployment in edge or resource-constrained IoT environments. As adversaries evolve and obfuscate their behaviors, these limitations highlight the need for more adaptive, interpretable, and lightweight anomaly detection methods to generalize beyond known threat signatures.

In contrast to existing literature, we propose a lightweight unsupervised method that outperforms supervised neural network approaches and runs 15 times faster on resource-constrained devices. The next section will

detail background information followed by our design section, and Sec. VI will compare our work with related methods, highlighting our novel contributions in efficiency and effectiveness.

IV. Background

A key challenge in cybersecurity is that encrypted network traffic prevents payload inspection. With over 70% of malware using encryption [10], detection instead relies on packet-level features (timing, size, metadata). We address this with an unsupervised method that combines PCA [4] for dimensionality reduction and the Elliptical Envelope [5] model for anomaly detection.

A. Principal Component Analysis

PCA is a statistical technique used to reduce the dimensionality of data by projecting data onto a lower-dimensional subspace while preserving as much variance as possible [6].

Let $X \in \mathbb{R}^{n\times d}$ be a dataset (in essence, a matrix). Dataset X contains n data samples $\{\mathbf{x}_1, ..., \mathbf{x}_n\}$, where each data sample $\mathbf{x}_i \in \mathbb{R}^d$ contains d features. PCA first computes the empirical mean vector as $\boldsymbol{\mu} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i$ and this mean is subtracted from each data point to obtain a zero-centered matrix as

$$X' = X - \mathbf{1}\boldsymbol{\mu}^\top \tag{1}$$

where $\mathbf{1} \in \mathbb{R}^{n\times 1}$ is a vector of ones.

Next, PCA computes the sample covariance matrix as

$$\Sigma = \frac{1}{n}X'^\top X' \in \mathbb{R}^{d\times d} \tag{2}$$

and performs eigenvalue decomposition on the covariance matrix as

$$\Sigma = V\Lambda V^\top \tag{3}$$

Here, $V \in \mathbb{R}^{d\times d}$ is a matrix of eigenvectors (principal directions), and $\Lambda \in \mathbb{R}^{d\times d}$ is a diagonal matrix containing the corresponding eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$, which represent the amount of variance captured by each direction.

To reduce the dimensionality, PCA selects the top k eigenvectors corresponding to the largest eigenvalues:

$$V_k = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k] \in \mathbb{R}^{d\times k}$$

The zero-centered matrix is then projected onto this k-dimensional subspace:

$$Z = X'V_k \in \mathbb{R}^{n\times k}$$

Each row $\mathbf{z}_i \in \mathbb{R}^k$ in Z is a lower-dimensional representation of its original data sample $\mathbf{x}_i \in \mathbb{R}^d$.

B. Elliptic Envelope

Elliptic Envelope is an unsupervised anomaly detection method that assumes normal data samples follow a multivariate Gaussian distribution. It identifies anomalies based on how far a given data sample deviates from the estimated distribution of the majority (benign) data [7].

Let the dataset X consist of n samples, where each data sample $\mathbf{x}_i \in \mathbb{R}^d$ represents a d-dimensional feature vector. In the training phase, Elliptic Envelope first calculates the empirical mean vector $\boldsymbol{\mu} \in \mathbb{R}^d$ as $\boldsymbol{\mu} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i$. This mean vector represents the estimated center (centroid) of the Gaussian distribution. Next, Elliptic Envelope computes the empirical covariance matrix $\Sigma \in \mathbb{R}^{d\times d}$ as below

$$\Sigma = \frac{1}{n}X'^\top X' \in \mathbb{R}^{d\times d} \tag{4}$$

During the detection phase, given a new data sample $\mathbf{x} \in \mathbb{R}^d$, mean vector $\boldsymbol{\mu}$ and covariance matrix Σ, Elliptic Envelope computes the Mahalanobis distance between this new data sample and the mean vector as below

$$d_M(\mathbf{x}, \boldsymbol{\mu}, \Sigma) = \sqrt{(\mathbf{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})} \tag{5}$$

This distance, in essence, measures how many standard deviations that data sample $\mathbf{x}$ lie from the center of the distribution, accounting for feature correlations.

Under the Gaussian assumption, the square of the Mahalanobis distance $d_M(\mathbf{x})^2$ approximately follows a chi-squared distribution with d degrees of freedom:

$$d_M(\mathbf{x})^2 \sim \chi^2_d \tag{6}$$

Elliptic Envelope selects a critical threshold δ from the chi-squared distribution such that if $d_M(\mathbf{x})^2 > \delta$, it outputs 1, which suggests data sample $\mathbf{x}$ is an anomaly sample (or an outlier). Otherwise, it outputs 0, indicating the data sample is benign (or an inlier).

V. Our Proposed Design

We develop an unsupervised malware detection approach that combines PCA with Elliptic Envelope. Specifically, PCA first reduces the dimensionality of extracted features from encrypted network traffic. Next, Elliptic Envelope is applied to identify outliers using the Mahalanobis distance [8] over the dimension space obtained by PCA. Our proposed detection method operates in three main phases: pre-processing, training, and detection.

Step 1: Pre-processing. Given a dataset of encrypted network packets, our method begins by extracting a set of statistical and protocol-level features from each packet. This pre-processing phase creates a structured feature matrix, where each row represents a single packet. This matrix serves as a numerical representation that is suitable for downstream analyses, such as dimensionality reduction and anomaly detection.

In this study, we extract 40 features per packet, capturing both low-level protocol indicators and higher-level traffic characteristics. These features include: $Header$ length, $Protocol$ type, $Time\_To\_Live$, $Rate$, fin_flag_number, syn_flag_number, rst_flag_number, psh_flag_number, ack_flag_number, ece_flag_number, cwr_flag_number, ack_count, syn_count, fin_count, rst_count, $HTTP$, $HTTPS$, DNS, $Telnet$, $SMTP$, SSH, IRC, TCP, UDP, $DHCP$, ARP, $ICMP$, $IGMP$, IPv, LLC, Tot_sum, Min, Max, AVG, Std, Tot_size, IAT, $Number$, and $Variance$.

Step 2: Training. We apply PCA to the extracted feature matrix to effectively reduce its dimensionality while preserving the majority of the variance present in the data. The lower-dimensional representation is then used to train an Elliptic Envelope model, which fits a multivariate Gaussian distribution to the benign data by accurately estimating its mean and covariance. These statistical parameters are crucial, as they form the core of our anomaly detection process, enabling us to identify deviations from normal behavior with high precision.

Step 3: Detection. During the detection phase, incoming packets are transformed using the same PCA projection and evaluated by the trained Elliptic Envelope model. The model assigns an anomaly score based on how much the packet deviates from the learned distribution. A packet is flagged as malicious if its score falls below a predefined threshold δ, which can be tuned based on the dataset. The packet is classified as benign if the score is above this threshold.

VI. Evaluation

A. Evaluation Metric

We assess the performance of our detection using precision, recall, F1-score, and ROC-AUC metrics defined below.

Precision: Measures the proportion of correctly predicted positive cases out of all predicted positives.

$$Precision = \frac{TP}{TP + FP}$$

Recall: Measures how many positive cases were correctly predicted.

$$Recall = \frac{TP}{TP + FN}$$

F1-score: The harmonic mean of Precision and Recall.

$$F_1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

Where:

- TP stands for **True Positives**: The number of correctly predicted positive instances.
- FP stands for **False Positives**: The number of negative instances incorrectly classified as positive.
- FN stands for **False Negatives**: The number of positive instances incorrectly classified as negative.

ROC-AUC (Receiver Operating Characteristic – Area Under the Curve) is a metric that evaluates how well a classifier distinguishes between positive and negative classes. The ROC curve plots the True Positive Rate against the False Positive Rate across various thresholds. The AUC value ranges from 0.5 (random guessing) to 1.0 (perfect classification), with higher scores reflecting stronger performance [23].

B. Dataset

We use a large-scale real-world dataset known as the *Canadian Institute of Cybersecurity Internet of Things (CIC IoT) Dataset 2023* [9] for our evaluation. This dataset consists of large merged CSV files that contain features extracted from extensive packet capture (PCAP) data. It includes 33 different attack types executed in an IoT topology comprising 105 devices. The dataset categorizes these attacks into seven major classes: DDoS (Distributed Denial of Service), DoS (Denial of Service), Reconnaissance (Recon), Web-based Attacks, Brute Force Attacks, Spoofing, and Mirai-based Attacks. Previous studies have widely used this dataset for anomaly detection over encrypted network traffic [2], [24].

C. Training and Testing

Our evaluation applies a 5-fold cross-validation scheme across five distinct files, from `merged01.csv` through `merged05.csv`, provided in the CIC IoT Dataset 2023 [9]. Each file contains a mixture of benign and anomalous network traffic. During each fold, one file is set aside for testing, while the other four are used for training, resulting in five independently trained models.

It is important to note that we only use samples labeled as benign for training to simulate a realistic unsupervised setting where anomaly labels are not available. Although we assume the training data is benign, this approach still relies on ground truth labels during the pre-processing phase to enforce the purity of the training set. In real-world scenarios, such labels are often unavailable, and we recognize the risk that traffic containing anonymous training data might degrade the performance or bias the learned baseline. Therefore, our evaluation represents the best-case scenario, and future work should consider methods that are robust and resilient to potential contamination in the training set.

Across the folds, typical packet counts per file are as follows:

- Benign packets used for training: $\sim$15,000
- Benign packets used for testing: $\sim$1,700
- Anomalous packets in testing: $\sim$340

This setup enables the evaluation of each model's ability to generalize from benign-only training data to unseen malicious behaviors during testing.

Evaluation Settings. We utilize a desktop with an Intel Core i7-9700 CPU (3.60 GHz), 16 GB RAM, and Windows 11. In addition, we also use a Raspberry Pi 5 with a Quad-core ARM Cortex-A76 CPU (2.4 GHz), 4 GB RAM, and Ubuntu 24.04.2. We implement our method in Python.

D. Experiments

Experiment 1: Performance Comparison with Recent Studies. To evaluate the the efficiency and effectiveness of our proposed method compared to existing literature, we replicate several prior studies, including: (1) a supervised learning approach using a Decision Tree [17], where both benign and malicious labels are used during training to fit a classification model; (2) an unsupervised anomaly detection method based on Deep AutoEncoders [25], where the Mateen method is trained on benign data and evaluated on sequential slices of test data using reconstruction error, with adaptive model updates triggered by performance drops or concept drift; and (3) a Graph Neural Network (GCN) model [19], which trains a supervised GNN on IoT packet features and constructs a k-nearest neighbor graph to define edge connectivity, enabling the model to perform binary classification of malicious activity.

We use the same dataset and a consistent data split across all methods to ensure a fair comparison. Specifically, we train each model using benign or labeled data according to its original design and evaluate it on a shared test set containing both benign and malicious packets. We utilize the publicly available source code from [25] and re-implement the methods from [17] and [19], as their code has not been made publicly available.

Table I summarizes our experimental results, showing the comparison with related studies. As the table illustrates, our method EVADE achieved the highest accuracy (98.6%), surpassing both supervised and unsupervised baselines. The results show that lightweight statistical models can outperform more complex learning approaches when combined with effective dimensionality reduction.

Experiment 2: Comparison with Unsupervised Learning. In this experiment, we compare the results of our method against several widely used unsupervised learning methods, including Isolation Forest, Local Outlier Factor (LOF), and One-Class Support Vector Machine (SVM). Specifically, we evaluate the training time, memory usage during training, and detection performance across these four methods on a desktop computer and a Raspberry Pi device.

As shown in Table IV, our method EVADE necessitates a longer training time and higher memory usage than other unsupervised methods. However, EVADE significantly outperforms these methods during the detection phase, as it operates much faster and consumes considerably less memory, as demonstrated in Table II and Table III.

Experiment 3: Visualization of our Detection Results. Fig. 1 illustrates the outlier detection capabilities of our EVADE framework, which utilizes an Elliptic Envelope model in a PCA-transformed space. The decision function generates a score for each test instance: higher values near zero (`red +`) suggest conformity with the training distribution. In contrast, lower (more negative) values (`blue +`) indicate anomalous behaviors as illustrated, most benign training samples (green circles) cluster within the red ellipse—the model's estimated boundary for inliers. Test instances outside this region, particularly those with lower decision scores, are identified as anomalies.

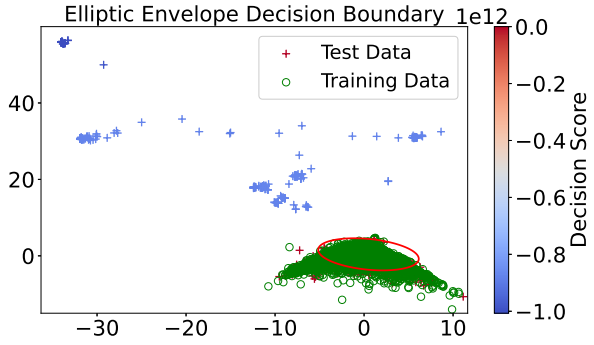

Fig. 1: EVADE: Elliptic Envelope Decision Boundary.

This visualization confirms that EVADE effectively distinguishes benign behavior from malicious outliers. We note that the decision threshold is selected based on the 15th percentile of the decision scores during evaluation. Future evaluations will investigate how changes to this threshold impact detection performance.

Experiment 4: Contamination Threshold In this experiment, we evaluated the sensitivity of our method to different contamination thresholds by systematically varying the contamination parameter used in the Elliptic Envelope model. Specifically, we assessed the detection performance, including precision, recall, F1-score, and ROC-AUC at four contamination levels: 0.5%, 10%, 15%, and 25%. This assessment allows us to quantify the trade-off between false positives and false negatives based on different assumptions about the proportion of anomalies in the network traffic.

Our findings reveal a consistently high ROC-AUC of 0.99, indicating that the algorithm reliably distinguishes between normal and anomalous traffic across the different contamination thresholds, as shown in Table V. A contamination of 15% achieves the best balance in this experiment in the trade-off between precision and recall sensitivity. Our EVADE method would benefit from incorporating adaptive thresholding or online contamination estimation

TABLE I: Comparison between our method and previous studies.

	Category	Algorithm	Feature Selection	Accuracy
[17]	Supervised	Decision Tree	PCA	96%
[19]	Supervised	GNN	Autoencoder	95.5%
[25]	Unsupervised	Mateen	Deep AutoEncoders	96.8%
EVADE (**Ours**)	Unsupervised	Elliptic Envelope	PCA	98.6%

TABLE II: Detection Performance (Desktop)

Algorithm	Precision	Recall	F1 Score	ROC-AUC	Detect Time (ms)	Memory (KB)
Elliptic Envelope	0.986	0.976	0.982	0.994	1.80	66
Isolation Forest	0.986	0.976	0.976	0.994	31.00	521
LOF	0.986	0.976	0.976	0.992	123.00	6,301
One-Class SVM	0.990	0.976	0.986	0.990	82.60	66

TABLE III: Detection Performance (Raspberry Pi 5)

Algorithm	Precision	Recall	F1 Score	ROC-AUC	Detect Time (ms)	Memory (KB)
Elliptic Envelope	0.988	0.978	0.988	0.992	2.74	66
Isolation Forest	0.986	0.976	0.978	0.994	47.41	526
LOF	0.986	0.976	0.976	0.992	347.47	6,314
One-Class SVM	0.990	0.976	0.986	0.990	86.22	66

TABLE IV: Training Time and Memory Usage (Desktop)

Model	Train Time (s)	Train Mem (KB)
Elliptic Envelope (Ours)	4.68	12,705
Isolation Forest	0.91	1,943
Local Outlier Factor	0.15	19,227
One-Class SVM	0.14	1,903

TABLE V: Contamination Thresholds

Contam. (%)	Precision	Recall	F1 Score	ROC-AUC
5	0.30	1.00	0.47	0.99
10	1.00	0.66	0.79	0.99
15	**0.99**	**0.98**	**0.99**	**0.99**
25	0.60	0.98	0.74	0.99

for practical and real-world applications to maintain its effectiveness in dynamic network conditions.

VII. Discussion and Limitations

We acknowledge two limitations of this study. First, as a statistical method, it may not adapt well if attackers employ AI-driven techniques to mimic benign traffic. Second, our evaluation relies on a fixed dataset, which limits generalizability. In future work, we plan to evaluate the method on more diverse datasets, such as CSE-CIC-IDS2018 with botnet traffic, and to integrate adaptive learning techniques such as incremental sliding windows, to improve robustness against evolving adversarial behaviors.

VIII. Conclusions

This paper presented a lightweight unsupervised learning approach that integrates Principal Component Analysis and Elliptic Envelope for anomaly detection. Using the CIC IoT Dataset 2023, our method extracts key network features, reduces dimensionality with PCA, and applies Mahalanobis distance-based anomaly detection to distinguish malicious activity from benign traffic. Compared to the existing literature, the experimental results demonstrated high precision and recall, reducing false positives and negatives while maintaining computational efficiency for real-time deployment. Our approach also improves network security by detecting zero-day threats without relying on labeled attack data, making it a scalable and adaptive solution for analyzing encrypted traffic.

Acknowledgments

This work was partially supported by the Ohio Department of Higher Education (ODHE) through the Strategic Ohio Council for Higher Education (SOCHE) and Ohio Cyber Range Institute (OCRI) sub-awards.

References

[1] R. Xie, J. Cao, E. Dong, M. Xu, K. Sun, Q. Li, L. Shen, and M. Zhang, "Rosetta: Enabling Robust TLS Encrypted Traffic Classification in Diverse Network Environments with TCP-Aware Traffic Augmentation," in *32nd USENIX Security Symposium (USENIX Security 23)*. Anaheim, CA: USENIX Association, 2023, pp. 625–642.

[2] N. Choudhury, D. Deka, A. Tewari, S. Gaur, S. Sarmah, V. Sharda, and S. Gautam, "Malicious Traffic Classification Using Convolutional Neural Network," in *2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, 2023, pp. 1–7.

[3] S. Han, Q. Wu, H. Zhang, and B. Qin, "Light-weight Unsupervised Anomaly Detection for Encrypted Malware Traffic," in *2022 7th IEEE International Conference on Data Science in Cyberspace (DSC)*. Los Alamitos, CA, USA: IEEE Computer Society, 2022, pp. 206–213.

[4] A. Maćkiewicz and W. Ratajczak, "Principal Components Analysis (PCA)," *Computers & Geosciences*, vol. 19, no. 3, pp. 303–342, 1993.

[5] S. Howard, "The Elliptical Envelope," 2007. [Online]. Available: https://arxiv.org/abs/math/0703048

[6] M. P. Deisenroth, A. A. Faisal, and C. S. Ong, *Mathematics for Machine Learning*. Cambridge: Cambridge University Press, 2021.

[7] J. M. Oliveira, J. Almeida, D. Macedo, and J. M. Nogueira, "Comparative Analysis of Unsupervised Machine Learning Algorithms for Anomaly Detection in Network Data," in *2023 IEEE Latin-American Conference on Communications (LATINCOM)*, 2023, pp. 1–6.

[8] Y. Dodge, "Mahalanobis distance," *The Concise Encyclopedia of Statistics, Springer, New York*, pp. 325–326, 2008.

[9] E. C. P. Neto, S. Dadkhah, R. Ferreira, A. Zohourian, R. Lu, and A. A. Ghorbani, "CICIoT2023: A real-time dataset and benchmark for large-scale attacks in IoT environment," *Journal of Sensors*, 2023.

[10] Cisco, "Secure network analytics – encrypted traffic analytics," https://www.cisco.com/site/us/en/learn/topics/security/what-is-network-traffic-analysis.html.

[11] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," in *Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP)*, Portugal, 2018.

[12] S. García, M. Grill, J. Stiborek, and A. Zunino, "An empirical comparison of botnet detection methods," *Computers & Security*, vol. 45, pp. 100–123, 2014.

[13] N. Moustafa and J. Slay, "Unsw-nb15: A comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)," in *Proceedings of the Military Communications and Information Systems Conference (MilCIS)*. IEEE, 2015, pp. 1–6.

[14] Verizon, "2023 data breach investigations report (dbir)," Verizon Enterprise Solutions, Tech. Rep., 2023. [Online]. Available: https://www.verizon.com/business/resources/reports/dbir/

[15] Cisco, "Cisco 2017 annual cybersecurity report," Cisco Systems, Tech. Rep., 2017. [Online]. Available: https://www.cisco.com/c/en/us/products/security/annual-cybersecurity-report.html

[16] Mandiant, "M-trends 2022 report," Mandiant (now part of Google Cloud), Tech. Rep., 2022. [Online]. Available: https://www.mandiant.com/resources/m-trends

[17] C. OKUR and M. DENER, "Detecting IoT Botnet Attacks Using Machine Learning Methods," in *2020 International Conference on Information Security and Cryptology (ISCTURKEY)*, 2020.

[18] X. Liu and Y. Cao, "Research on Network Intrusion Detection Techniques Based on Feature Selection Model and Recurrent Neural Network," in *Proceedings of the 8th International Conference on Cyber Security and Information Engineering*, ser. ICCSIE '23. New York, NY, USA: Association for Computing Machinery, 2023, pp. 45–51.

[19] Z. Sun, A. M. Teixeira, and S. Toor, "GNN-IDS: Graph Neural Network based Intrusion Detection System," in *Proceedings of the 19th International Conference on Availability, Reliability and Security*, ser. ARES'24. New York, NY, USA: Association for Computing Machinery, 2024.

[20] F. Deldar and M. Abadi, "Deep Learning for Zero-day Malware Detection and Classification: A Survey," *ACM Computing Surveys*, vol. 56, no. 2, 2023.

[21] I. J. King and H. H. Huang, "Euler: Detecting Network Lateral Movement via Scalable Temporal Link Prediction," *ACM Trans. Priv. Secur.*, vol. 26, no. 3, 2023.

[22] S. Dadkhah, H. Mahdikhani, P. K. Danso, A. Zohourian, K. A. Truong, and A. A. Ghorbani, "Towards the Development of a Realistic Multidimensional IoT Profiling Dataset," in *Proceedings of the 19th Annual International Conference on Privacy, Security & Trust (PST)*, Fredericton, Canada, 2022.

[23] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, 2006.

[24] Q. Wang, M. Xie, Z. Wu, and D. Yang, "Network intrusion detection and dynamic defense method based on unsupervised machine learning," in *Proceedings of the 2023 International Conference on Computer Simulation and Modeling, Information Security (CSMIS)*, 2023, pp. 75–80.

[25] F. Alotaibi and S. Maffeis, "Mateen: Adaptive Ensemble Learning for Network Anomaly Detection," in *Proceedings of the 27th International Symposium on Research in Attacks, Intrusions and Defenses*, ser. RAID'24. New York, NY, USA: Association for Computing Machinery, 2024, pp. 215–234.