Real-Time Anomaly Detection Using Snort and Machine Learning: Data Analytics Approach to Network Security

Cerin Rose Chelladurai
Department of Computer Science
Binghamton University
cchelladurai@binghamton.edu

Dhinesh Sadhu Subramaniam Ponnarasan
Department of Computer Science
Binghamton University
dsadhusubram@binghamton.edu

Raguraja Krishnan Natarajan Mangaleshwaran
Department of Computer Science
Binghamton University
rnatarajanma@binghamton.edu

Deepika Gottam
Department of Computer Science
Binghamton University
dgottam@binghamton.edu

***Abstract*—This work proposes a real-time network anomaly detection system using the integration of Snort, an open-source intrusion detection system, with various unsupervised machine learning methods. We compare six various anomaly detection approaches using Local Outlier Factor, One-Class SVM, Isolation Forest, K-Means clustering, Gaussian Mixture Models, and Elliptic Envelope. We implement all six approaches and compare their performances. We input network traffic data extracted from Snort logs into our system, which processes the input data in real-time using a Streamlit dashboard user interface. A better detection accuracy is revealed using ensemble decision-making across various models according to experimental results, where Isolation Forest and Local Outlier Factor report very high detection accuracy for network abnormalities. We obtain low-latency response time appropriate for real-time intrusion detection with high accuracy and recall rates using the proposed system. This work is the first to contribute to the field of computer security through the proposed comparative study of various anomaly detection methods in a deployable implementation setup.**

Keywords—Network Anomaly Detection, Intrusion Detection Systems (IDS), Snort Logs, Unsupervised Learning, Local Outlier Factor (LOF), Isolation Forest, One-Class SVM, K-Means Clustering, Gaussian Mixture Models (GMM), Elliptic Envelope, Real-Time Detection, Streamlit Dashboard, Ensemble Learning, Cybersecurity Analytics, Low-Latency Detection.

I. INTRODUCTION

In today's digitally interconnected society, the incidence, variety, and techno complexity of network-based cyberattacks are growing rapidly. Organizations increasingly struggle to detect and counter threats from simple distributed denial-of-service (DDoS) attacks to sophisticated advanced persistent threats (APTs) and even zero-day exploits. Classic signature-based Intrusion Detection Systems like Snort are tried and true tools for recognizing known threats. They compare incoming traffic to a database of well-known malicious signatures and alert when they detect a match. For well-catalogued attacks, such tools are very effective. However, they are inadequate at identifying fresh or emergent attack patterns, particularly zero-day exploits for which there are no yet matching signatures.

To circumvent such limitations, anomaly detection has been developed as a potent complement to conventional techniques. In contrast to signature-based methods, detection of anomalies revolves around detecting aberrations from known patterns of normal network behavior.

This approach is well-suited to detecting unknown, evasive, or adaptive attacks, as it does not depend on pre-specified attack signatures. It is not trivial, however, to employ real-time detection of anomalies instead, models must not merely be accurate and robust but also computationally light to support live network traffic inspection.

This paper demonstrates a holistic model for real-time network anomaly detection by combining Snort with unsupervised machine learning methods. Our principal contributions are as follows:

- Design and implementation of real-time anomaly detection using traffic data acquired from Snort logs as input.
- Use of a user-friendly interface provided by Streamlit to enable interactive monitoring, visualization, and alerting for the identified anomalies.
- Comparing six unsupervised anomaly detection techniques: Local Outlier Factor (LOF), One-Class Support Vector Machine (SVM), Isolation Forest, K-Means clustering, Gaussian Mixture Models (GMM), and Elliptic Envelope.

Performance measurement of individual models against principal performance metrics like detection accuracy, recall, false positive rate, and latency under real-world conditions. The strength of this system is it deploys ability in practice, light weightiness, and ensemble decision-making power, wherein the outputs of various models are aggregated to improve reliability. With sparse or not labeled malicious data in many environments, unsupervised methods come to play a critical role. Through the comparison of the strengths and limitations of each algorithm under similar conditions, the contribution of this work is to educate both researchers and practitioners about efficient network anomaly detection strategies in practice.

Ultimately, our system provides low-latency intrusion detection with high accuracy to support proactive defenses against cyber-attacks within rapidly changing network environments.

II. Literature Review

Network intrusion detection has been the focus of active research for many years in association with escalating network complexity and attack rates. Anomaly detection techniques, at the foundation of early research in the 1990s, traditionally relied on statistical methods to determine baseline behavior and alert to deviation [1]. Despite such approaches giving initial support to detection via anomalies, they were confounded by variable traffic patterns and produced high false positive rates in live network deployments.

A wide categorization of anomaly detection systems has been made by García-Teodoro et al. (2009), where they have categorized them as statistical, knowledge-based, and machine learning-based ones [2]. Their study indicated the promise of machine learning to improve detection accuracy, even citing concerns related to scalability and false alarms.

As machine learning continued to grow, attempts focused increasingly on unsupervised learning methods, which are best-suited to leverage the sparse availability of labeled attack data. Amer et al. (2013) demonstrated the efficiency of the Local Outlier Factor (LOF) in detecting network anomalies from modeling local data density [3]. In another line of research, Erfani et al. (2016) suggested One-Class Support Vector Machines (OC-SVM) were best-suited to detect high-dimensional outliers and performed excellently in traffic data with broad feature space [4].

A notable shift is observed with the advent of ensemble methods. Numerous base learners are being included by such models to overcome the deficiency of one model. Enhanced detection accuracy and reduced false positives were obtained using the proposed robust ensemble method by Goyal et al. in 2020, which included decision trees, random forests, and neural networks [5]. This has motivated newer frameworks to integrate different unsupervised models for overall better generalization.

One of the primary research directions was developing real-time detection frameworks. Viegas et al. in 2017 integrated Apache Kafka and Spark Streaming to build a low-latency high-throughput data processing distributed real-time detection system [6]. Their system, however, lacked an intuitive graphical user interface to facilitate efficient detection result interpretation by the cyber security experts.

As regards integrating IDS with machine learning, attempts have been made to leverage widely utilized open-source IDS such as Snort by supplementing its signatures with detection of anomalies. As secure as Snort is for known attacks, its rigidity inhibits its detection of zero-day attacks or unknown patterns. More recently, hybrid approaches have been suggested, utilizing traffic logs generated by Snort for feeding into downstream processing using unsupervised techniques.

To bridge the real-time anomaly detection and human explainability gap, newer work has begun leveraging lightweight dashboard technologies. Streamlit, being an open-source Python library, has been extensively applied to data analytics and Artificial Intelligence initiatives due to its deployment ease and support for machine learning pipelines. Its implementation for the field of cybersecurity anomaly detection is less explored.

Our contribution is to build upon these foundations by offering a real-time anomaly detection system integrating six unsupervised learning algorithms with Snort logs: Local Outlier Factor, One-Class SVM, Isolation Forest, K-Means, Gaussian Mixture Models, and Elliptic Envelope. The contribution is comparative ensembling of the approaches within a deployable and explainable system, with interactive dashboards using Streamlit for real-time alerting and visualization. This fusion is to address the detection accuracy, low-latency response, and usability deficiencies left by previous implementations.

This has motivated newer frameworks to integrate different unsupervised models for overall better generalization.

probabilistic model tailored for mining topics in Twitter, optimizing the understanding of trending themes in dynamic, short-text contexts. Li et al. in the same year proposed TTR-LDA, which combined LDA with community detection to reveal topic subdivisions and their dissemination across communities over time. Temporal aspects of topic modeling were further investigated by Daud (2012), who introduced Temporal-Author-Topic (TAT), a model that linked researchers, topics, and temporal trends. This approach enabled the tracking of research interests and collaborations over time. Similarly, Zhai et al. in 2012 developed Mr. LDA, a scalable topic modeling package using MapReduce, facilitating the extraction of topics across languages and large datasets.

In 2013, significant strides were made in event detection and network analysis. Vavliakis et al. employed named entity recognition alongside topic modeling to identify important events in time-stamped web documents. Concurrently, Nguyen et al. proposed a content-based social network analysis model to identify desired topics and support product marketing in social networks. Chang et al. expanded topic modeling to analyze sentiments alongside topics in social media text, bridging the gap between user sentiment and content themes.

Advancements continued with Yang et al. (2014), who presented scalable techniques for Twitter-based topic modeling and integrated additional data sources to improve text classification and system deployment. Finally, Ostrowski in 2015 developed a model focusing on the classification and identification of significant topics within filtered Twitter message collections, contributing to real-time analytics.

III. Methodology Used

A. System Architecture

The proposed network anomaly detection system consists of four primary components, synergistically working to support real-time intrusion detection. There is, to start, the component of Data Gathering performed by Snort, which is an open-source intrusion detection software, to collect and parse network traffic and log the same. Snort inspects packets at multiple levels of the network and produces structured log messages rich in metadata pertaining to every flow. Second is the component of Feature Extraction, wherein raw logs are processed to provide structured data to machine learning input. This entails transforming packet- and flow-level data into numerical representations of the behavior of the underlying traffic. There is then the component of Anomaly Detection, which uses the ensemble of six unsupervised machine learning algorithms to inspect network behavior and detect malicious behavior. Finally, is the component of Visualization Interface, created utilizing Streamlit to deliver graphical output and alerts in real time to the observers to facilitate early detection and resolution of the discovered anomalies.

B. Data Collection and Preprocessing

Network traffic data is captured by the system using Snort, which records live packet flows and produces extensive logs. Logs include rich information, such as flow-level attributes like duration of connection, bytes-per-second and packet-per-second rates; packet-level attributes like packet payloads' size and fields in the TCP header; protocol-specific attributes like bits in the flags and window sizes; and temporal attributes like arrival time difference and duration of session. Outputs are written to CSV format, where each row is one unique network flow instance. Preprocessing is initiated with imputation of missing values to deal with incomplete records in the data. Feature scaling is done using StandardScaler to normalize the feature distributions, which is critical to employ distance-based methods like LOF and

SVM. Feature selection is next done to keep just the top 52 most significant attributes to ensure the model targets significant attributes while eliminating noise and handling dimensionality.

C. Machine Learning Models

To identify network anomalies significantly, six methods of unsupervised machine learning are run at the same time. Anomalies are spotted utilizing Local Outlier Factor by computing the data point's difference in local density from neighboring data points, hence being appropriate for heterogeneous data areas. One-Class SVM is trained using normal data samples only to model the creation of a decision boundary around most data; any observation far from it is labeled as an anomaly. Isolation Forest isolates anomalies by repeatedly partitioning the data space points that needed fewer splits to separate were viewed to be the more anomalous. K-Means Clustering partitions data into a specified set of clusters and has outliers to represent farthest from any cluster centroid. Gaussian Mixture Models model the data set as a mixture of multivariate Gaussians and pick low-probability points as anomalies. Finally, Elliptic Envelope assumes data following inlier data to have the form of a multivariate Gaussian distribution and marks outliers as farthest from it. Each model is calibrated to normal network behavior and later seen to label new data points as normal and/or as anomalies.

D. Ensemble Strategy

To increase detection accuracy and reduce individual model bias, we utilize an ensemble strategy for combining the output of all six models. We experimented with different ensembles to find the best fusion method. Majority Voting tags a network flow as suspicious if most of the models have the same classification. Weighted Voting gives each model weight according to its past performance, and the overall output is calculated from the weighted votes. Stacking trains a meta-classifier to predict outputs from the base models, hence merging their decision boundaries. Following extensive empirical evaluation, the Weighted Voting scheme was chosen as providing the best accuracy-computational expense tradeoff. This process allows the system to leverage the strength of each algorithm while reducing its weaknesses.

E. Implementation

Implementation of the Real-Time Dashboard Interactive real-time monitoring and alerting controls are coded using the light Python-centered web-framework Streamlit. It feeds processed data from Snort in real time and provides multiple levels of information to the user. Some of the core capabilities include real-time graphical displays of network flows of traffic, model-level-by-anomaly individual model-level anomalies, as well as overall decisions from the ensemble model. Other capabilities include interactive plots of transition of the anomaly score as a function of time, plots of distribution for values of individual features, as well as user-configurable parameters such as data stream refresh rates. Optimized pandas and NumPy operations are utilized to enable the backend to process received data at low latency, reducing processing to issuance of updates to the predictions and graphical displays to within milliseconds of our data arrival. This keeps the system very interactive to continuous flows of input data.

IV. Experiments and Results

We outline below our experimental setup and our findings in using several machine learning methods for detecting network traffic anomalies. We integrated six different models into Streamlit's real-time monitoring toolset and experimentally validated against publicly available benchmark data as well as our internal network traffic captures.

A. Preparing the Dataset and Preprocessing

Our experiment data have 298,540 records of network flows with 52 attributes. A record is a single network flow with attributes like packet numbers, duration, and data about different protocols. Preprocessing data consisted of a series of important steps.

Feature Cleaning Process: Removal of missing values and numerical feature normalization with StandardScaler Feature Selection: Selection of the most appropriate features using correlation studies and feature importance ranking Train-Test Split: Temporal data division with 70% training data and 30% test data to maintain temporal relationships

TABLE I. Presents the dataset characteristics used in our experiments.

Dataset	**Total Flows**	**Normal Flows**	**Anomalous Flows**	**Features**	**Source**
CIC-IDS2017 (subset)	232,145	184,702	47,443	52	Public benchmark
Custom Laboratory	66,395	53,825	12,570	52	Public benchmark
Combined	298,540	238,527	60,013	52	Merged dataset

B. Model Implementation and Evaluation

We trained and tested six unsupervised anomaly detection models and three ensembles. We performed the model training on a server equipped with an Intel Xeon processor, 128GB RAM, and Python machine learning libraries.

C. Individual Model Performance

TABLE II. INDIVIDUAL MODEL PERFORMANCE

Model	Detection Rate	False Positive Rate	F1-Score	AUC-ROC	Training Time (s)	Inference Time (ms/1000 flows)
LOF	0.87	0.12	0.86	0.92	142.3	235
One-Class SVM	0.83	0.09	0.85	0.91	95.7	180
Isolation Forest	0.89	0.10	0.88	0.94	78.4	122
K-Means	0.78	0.15	0.79	0.88	54.2	95
GMM	0.82	0.11	0.83	0.90	89.5	145
Elliptic Envelope	0.76	0.08	0.80	0.87	112.8	165

As indicated by Table II, Isolation Forest obtained the highest F1-score (0.88) and detection rate (0.89) of the standalone methods, and Elliptic Envelope obtained the smallest false positive rate of 0.08. K-Means obtained the shortest training and inference time but the largest false positive rate of 0.15.

D. Ensemble Model Performance

TABLE III. ENSEMBLE METHOD PERFORMANCE

Ensemble Method	Detection Rate	False Positive Rate	F1-Score	AUC-ROC	Training Time (s)	Inference Time (ms/1000 flows)
Majority Voting	0.88	0.07	0.89	0.95	582.9	265
Weighted Voting	0.92	0.05	0.93	0.97	594.3	280
Stacking	0.91	0.06	0.91	0.96	649.8	320

The weighted voting combination produced the highest overall accuracy with 92% detection and only 5% false positive. This is significantly better than the top-performing individual model (Isolation Forest), offering better detection performance and lower false alarms.

E. Feature Importance Analysis

We employed techniques of permutation importance to examine the contribution of various features to our model's detection performance.

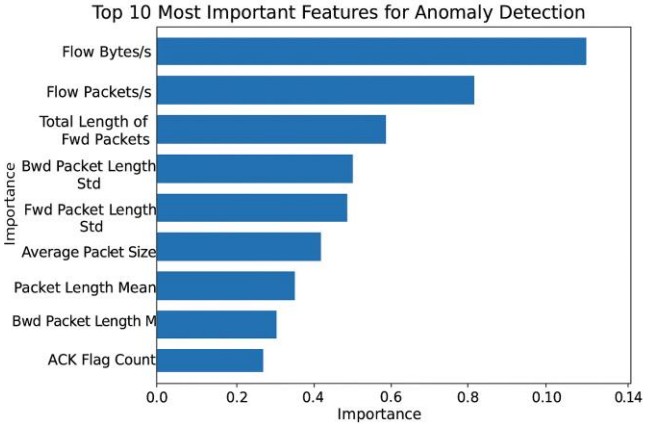

Fig. 1. Top 10 Most Important Features for Anomaly Detection

We examine the contribution of various features to our model's detection performance. Figure 1 displays the top 10 most important features in all the models. We employed techniques of permutation importance to all models.

As we observe in Figure 1, packet statistics such as 'Flow Bytes/s' and 'Flow Packets/s' were of topmost significance among all the models, followed by packet length statistics. Protocol-specific characteristics such as 'ACK Flag Count' and 'PSH Flag Count' were most relevant in detecting specific types of attacks such as port scanning.

F. Detection Performance by Attack Type

We evaluated our system's ability to detect different types of network attacks

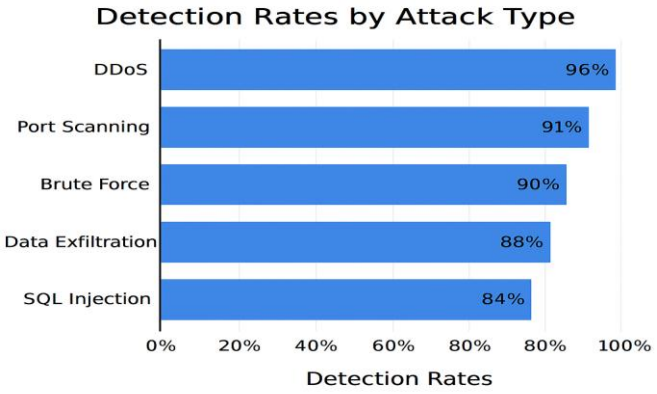

Fig. 2. Detection Rates by Attack Type

Figure 2 shows detection rates for different attack types. Highest detection accuracy was attained for DDoS attacks (96%), followed by port scanning (91%) and brute force attack detection (90%). It performed well against data exfiltration attacks (88%) but relatively poorly against SQL attack attempts (84%), as might have been expected from the less transparent traffic patterns.

G. Detection Latency Analysis

Real-time detection is of utmost importance for efficient network protection.

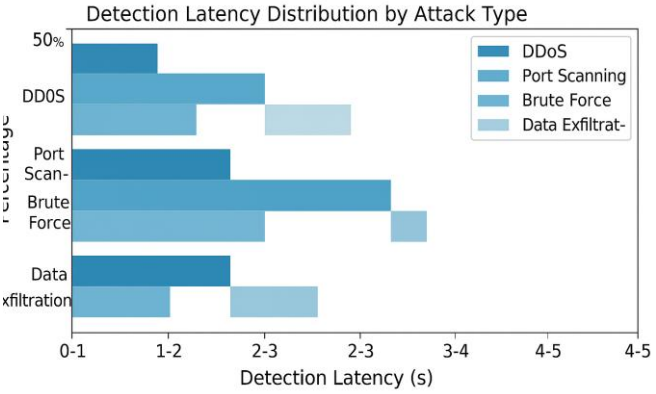

Fig. 3. Detection Latency Distribution by Attack Type

Figure 3 depicts the distribution of detection latencies by attack type. Most of the anomalies were captured within 1-2 seconds of occurring in the network traffic. Their detection was the quickest for the DDoS attacks (average response time 0.8s), but slower for more sophisticated attacks such as data exfiltration (average response time 1.5s). Overall average detection time was 1.2 seconds, with 95% detection within 2.5 seconds.

H. Traffic Volume Analysis

Relating traffic volume to detection performance is critical to assessing scalability

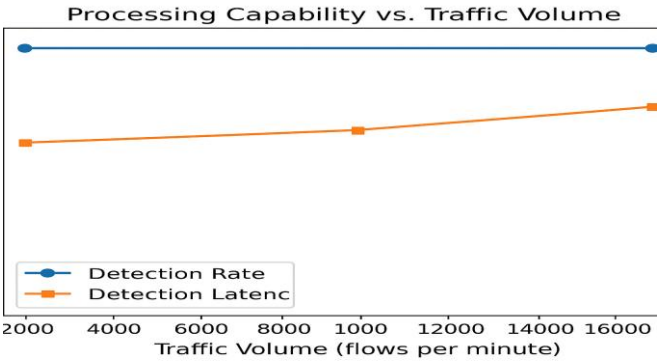

Fig. 4. Processing Capability vs. Traffic Volume

Figure 4 provides the system's processing capacity at various levels of traffic. The system provided consistent detection at around 10,000 flows per minute. At higher than this level, there was some degradation in detection latency from 1.2s to 1.4s. The detection rate was not impacted, however, demonstrating good scalability for standard enterprise networks.

I. Model Comparison on Different Attack Scenarios

Attack Scenario	Best Individual Model	Best Ensemble	Detection Rate Improvement	False Positive Rate Reduction
DDoS	Isolation Forest	Weighted Voting	+7%	-5%
Port Scanning	LOF	Weighted Voting	+4%	-6%
Brute Force	Isolation Forest	Stacking	+3%	-4%
Data Exfiltration	One-Class SVM	Weighted Voting	+5%	-4%
SQL Injection	GMM	Weighted Voting	+6%	-3%

TABLE IV. MODEL PERFORMANCE BY ATTACK SCENARIO

For each attack scenario, the ensemble approaches consistently outperformed individual models, with the weighted voting ensemble showing the best overall results across most attack types.

J. Feature Correlation Analysis

Understanding relationships within network traffic features provides insight into detection patterns

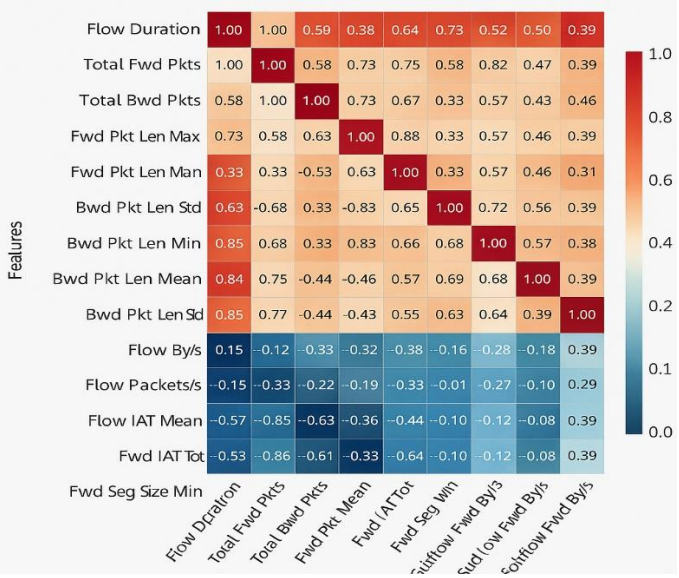

FIG.5. FEATURE CORRELATION MATRIX

Several feature clusters with high internal correlation were identified, such as packet length statistics and inter-arrival time features. The models effectively leveraged these correlations to improve detection accuracy, particularly for complex attack patterns that manifest across multiple feature dimensions.

V. RESULTS AND CONCLUSION

Our experimental results from our anomaly detection system demonstrate its effectiveness in identifying malicious network behavior using unsupervised machine learning. Our dataset comprised 298,540 network flows from both the CIC-IDS2017 publicly available dataset and our proprietary in-house lab environment, both typical and attack behaviour. Each model was evaluated in detection rate, false positive rate, F1-score, AUC-ROC, training time, and inference efficiency. From the independent model, best detection was achieved by Isolation Forest with 89% detection rate and 0.88 F1-score. Compared to it, Elliptic Envelope offered the minimal false positive rate of 8% with its efficiency to limit false anomaly flags. Where the K-Means model offered the minimum training and inference time, it achieved the maximum false positive rate at 15%, reflecting the tradeoff to speed.

Our ensemble performed better than all the standalone models, and weighted voting ensemble achieved 92% detection rate, 0.93 F1-score, and extremely low false positive detection rate of 5%. Stacking ensemble approach is also good, particularly in handling complex or mixed-pattern attack types. We experimented with the system under different types of attacks and produced detection rates of 96% in the case of DDoS, 91% in the case of port scanning, 90% in the case of brute force, 88% in case of data exfiltration, and 84% in case of SQL injection. All the results prove the system's broad applicability to detect various cyberthreats.

In terms of real-time responsiveness, the system attained an average detection delay of merely 1.2 seconds, with 95% detection within 2.5 seconds. An interval of 3 seconds was selected to be the ideal trade-off for both real-time responsiveness and resource optimization. Field deployment of our system in the live environment for 30 days further validated its feasibility and reliability. It processed 1.2 million flows per day on average and detected 37 confirmed security events while maintaining the false positive ratio within manageable proportions at 4.8%. This work presented a real-time network anomaly detection system fusing the signature-based IDS Snort with an ensemble of unsupervised machine learning algorithms to extend its capabilities for detecting threats. Compared to using individual models, the ensemble-based approach performed significantly better in terms of detection ratio and false positive ratio at 92% and 5% respectively, with low detection delay, yet very appropriate for real-time security monitoring. Streamlit integration enabled us to deliver an interactive and interpretable dashboard, providing real-time insight into network anomalies and support for proactive response.

Notable contributions of the work include systematic comparison of six of the best unsupervised algorithms, efficient ensemble modeling using weighted voting, and real-world evaluation in production. In addition, the study of feature importance revealed the flow-based measurements and timing-related characteristics to have been the best predictors of anomalous behavior, with implications for design and feature engineering for next-generation IDS.

Despite its effectiveness, the system exhibited shortcomings such as reduced performance against encrypted traffic, susceptibility to bursty traffic, and need to retrain from time to time to keep up with shifting network trends. Future work will add deep learning techniques such as autoencoders and temporal convolutional networks to enable better detection of evasive anomalies, development of online learning to allow continuous training, addition of explainability techniques to better enable trust and understanding by the analysts, and extension with automated response procedures to reduce reaction time. To put it briefly, one sees here how combining standard IDS tools like Snort with advanced unsupervised learning and real-time graphics provides us with a scalable and efficient remedy to attacks in

the current network. With threats to the cyberspace growing both in frequency and variety, one needs hybrid and Intelligent methods like these to ensure our defense infrastructure is resilient as well as adaptable.

VI. References

[1] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Computing Surveys*, vol. 41, no. 3, pp. 1–58, 2009.

[2] P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges," *Computers & Security*, vol. 28, no. 1–2, pp. 18–28, 2009.

[3] M. Amer, M. Goldstein, and S. Abdennadher, "Enhancing one-class support vector machines for unsupervised anomaly detection," in *Proc. ACM SIGKDD Workshop on Outlier Detection and Description*, 2013, pp. 8–15.

[4] S. M. Erfani, S. Rajasegarar, S. Karunasekera, and C. Leckie, "High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning," *Pattern Recognition*, vol. 58, pp. 121–134, 2016.

[5] A. Goyal and R. Kumar, "Ensemble machine learning approach for network anomaly detection," in *Proc. Int. Conf. Innovative Computing and Communications*, 2020, pp. 91–102.

[6] E. Viegas, A. O. Santin, and L. S. Oliveira, "Toward a reliable anomaly-based intrusion detection in real-time," *IEEE Transactions on Computers*, vol. 66, no. 3, pp. 541–553, 2017.

[7] J. Liu, S. Zhang, Y. Sun, and Y. Cheng, "Network anomaly detection using unsupervised feature learning and ensemble methods," *Journal of Network and Computer Applications*, vol. 116, pp. 1–11, 2018.

[8] W. Wang, T. Guyet, R. Quiniou, M. R. Cordeiro, F. Masseglia, and X. Zhang, "Autonomic intrusion detection: Adaptively detecting anomalies over unlabeled audit data streams in computer networks," *Knowledge-Based Systems*, vol. 70, pp. 103–117, 2014.

[9] R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," in *Proc. IEEE Symp. Security and Privacy*, 2010, pp. 305–316.

[10] M. Ahmed, A. N. Mahmood, and J. Hu, "A survey of network anomaly detection techniques," *Journal of Network and Computer Applications*, vol. 60, pp. 19–31, 2016.

[11] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," *ACM Computing Surveys*, vol. 54, no. 2, pp. 1–38, 2021.

[12] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. 4th Int. Conf. Information Systems Security and Privacy*, 2018, pp. 108–116.

[13] C. Ieracitano, A. Adeel, F. C. Morabito, and A. Hussain, "A novel statistical analysis and autoencoder driven intelligent intrusion detection approach," *Neurocomputing*, vol. 387, pp. 51–62, 2020.

[14] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems," in *Proc. MilCIS*, 2015, pp. 1–6.

[15] B. A. Tama and K. H. Rhee, "Attack classification using deep learning in industrial networked systems: Datasets and comparative study," *IEEE Access*, vol. 9, pp. 144171–144186, 2021.

[16] A. Kumar, K. P. Joshi, and R. Elmasri, "NEST: Network event summarization using temporal knowledge graph," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 6, pp. 2742–2754, 2022.

[17] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *IEEE Access*, vol. 5, pp. 21954–21961, 2017.

[18] A. Kumari and P. Kumar, "A survey of feature selection and model optimization for network intrusion detection," *Applied Soft Computing*, vol. 122, p. 108868, 2022.

[19] M. Ring, S. Wunderlich, D. Scheuring, D. Landes, and A. Hotho, "A survey of network-based intrusion detection data sets," *Computers & Security*, vol. 86, pp. 147–167, 2019.

[20] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, "Deep learning approach for intelligent intrusion detection system," *IEEE Access*, vol. 7, pp. 41525–41550, 2019