Volume 4 Issue VII, August 2025, No. 70, pp. 894-909
Submitted 22/6/2025; Final peer review 28/7/2025
Online Publication 2/8/2025
Available Online at http://www.ijortacs.com

A DEEP LEARNING FRAMEWORK FOR REAL-TIME CYBER THREAT DETECTION AND MITIGATION IN NETWORKED ENVIRONMENTS

[1*]Ezeh Ebere M., [1]Asogwa T.C.

[1,1*,2] Computer Science Department, Enugu State University of Science and Technology

Email: [1*]ebydent@gmail.com [1]tochukwu.asogwa@esut.edu.ng

Abstract

The changing nature of the frequency and complexity of cyber threats necessitates the need to implement network security based on intelligent and adaptive security solutions with the ability of detecting and mitigating threats on a real-time basis. This paper proposes an end-to-end framework of cyber threat feature evaluation and mitigation which blends an auto encoder-based hybrid deep learning model, Long Short-Term Memory and Autoencoder (LSTM+AE) with cross correlation-based feature extraction algorithm. The suggested system will dynamically examine incoming network traffic, isolate high-affect features, characterize dangers in real time and apply instant mitigation interventions at transport layer through Transmission Control Protocol (TCP) controls. Context-aware threat attribution techniques are provided by matching network activity to user logs, to improve traceability and responsiveness precision. It has been implemented in Python, with libraries TensorFlow, Keras and Scikit-learn, and the experimental application was tested on the NSK-DD dataset in a virtualized testbed. The experimental accuracy of detection (98.6%), precision (97.9%), recall (98.1%), and the F1-score (98.0%) were high along with the average mitigation latency of less than 1.5 seconds and a rate of false positive of 1.2 percent. Besides, generalizability of the framework was confirmed by protocol-agnostic analysis over TCP, UDP, and ICMP streams. The findings support the system to be effective in improving the resilience of cybersecurity by facilitating pro-active, smart, and efficient management of threats in the complex networked setting.

Keywords: Cybersecurity; Threat Mitigation; LSTM-Autoencoder; Real-Time Detection; Cross-Correlation, Feature Extraction

1. INTRODUCTION

Network technologies have been increasing the overall quality of services over the past few decades, but they have also made network security more difficult (Elberri et al., 2024). Threat vectors like denial of service, malicious insider threats, man-in-the-middle attacks, and phishing attacks are a few examples of common attack types used by network intruders to illegally attack network environments and violate data integrity, availability, and confidentiality. As a result, feature assessment models that look into packet features' characteristics towards network environments and categorise threats are necessary (Ren et al., 2020).

Historically, a number of technologies have been used to counter threats, including firewalls, intrusion detection systems, and antivirus software (Ahmed et al., 2022). However, these current solutions are unable to

Corresponding Author Tel: +2348037661660

provide the dependable security standards required to restore user confidentiality and network integrity due to the complex and unexpected nature of these threat vectors (Abdulrahman et al., 2023). There are advantages and disadvantages to technological advancements. It has remained a research hotspot in the cyber scientific community, and one of its drawbacks is that threat attackers use it to optimise threat characteristics and create new threat vectors that are very challenging to identify with the security measures in place today (Celebi et al., 2023).

A more modern method for real-time threat classification is feature evaluation, which looks into the characteristics of data packets. Game theory, machine learning, fuzzy logic, encryption, and other optimisation techniques have recently dominated studies due to this approach's effectiveness in differentiating features of potential threats from typical legitimate features (Kim et al., 2020; Parra et al., 2019; Li et al., 2020; Rouamel et al., 2022; Ren et al., 2020; Piazza, 2020; Pawlick et al., 2019; Ferguson et al., 2019). Machine Learning (ML) stands out as the most effective of these methods because it can learn directly from data, model the issue, and then utilise the reference information for feature evaluation to categorise the danger.

Alkhalidi and Yaseen (2021), who used a semi-supervised machine learning approach, are among the articles that used machine learning for network feature assessment. In order to create feature assessment models and compare threat detection, Ghosh et al. (2019) experimented with Support Vector Machine (S-VM), Neural Network, Bayes Classifier, and Decision Tree, respectively. Although machine learning (ML) may accurately categorise packet attributes to identify threats, Elberri et al. (2024) found that deep learning offers an even better answer than ML approaches.

Compared to machine learning (ML), deep learning (DL), a multi-layer convolutional neural network, offers several benefits, including increased accuracy, automated feature extraction, more reliable feature selection, etc. A real-time cyber threat feature assessment and mitigation framework using cross-correlated deep learning techniques is required because studies like Sun et al. (2022), Almousa et al. (2022), and Sharma et al. (2022) have all used DL to manage cybersecurity challenges and have shown good performances. However, Guo et al. (2023) and Elberri et al. (2024) have shown that DL is prone to the problem of over-fitting and false positive results. To help identify complex features of cyber threats and legitimate packets, the cross-correlated approach is designed for quality assurance in the feature extraction process. These features are then fed into an enhanced deep learning algorithm that uses an encoded convolutional neural network for real-time threat classification, feature identification, and concatenation.

2. METHODOLOGY

The feature-driven development approach is the methodology employed in this study. Because it stresses an iterative, user-centred process that is perfect for handling complicated cybersecurity issues, the approach is the optimal methodology for creating a deep learning feature assessment model for threat mitigation. This method

makes it possible to create more useful and flexible solutions by emphasising an awareness of users' requirements and the ever-changing nature of threats. Design thinking's iterative process enables ongoing testing and improvement, guaranteeing that the deep learning models are both capable of identifying hazards and adaptable enough to reduce emerging or changing risks. This makes it especially appropriate for a field where user contact is essential for real-world applications and threats are ever-evolving.

2.1 Data Collection

The data used in this study was collected from the NSK-DD dataset on Kaggle. NSK-DD provides detailed network traffic records, including normal and attack traffic, making it a reliable source for intrusion detection research. The collected data included 16 features such as source and destination ports, IP addresses, protocol types, packet counts, and statistical flow characteristics. These features were essential in capturing the distinctions between normal and malicious traffic, aiding in the development of an accurate detection model. The sample size of the data collected is 404289 records of network information. This made up the secondary data source, while the primary data source contained similar network attributes, but the test-bed is the National Cyber Security Coordination Center (NCCC), Headquarters, Three Arms Zone, and Abuja, Nigeria. The sample size of data collected is 304333 records of network behaviour, which constitutes both normal and abnormal network information. The total sample size of data collected is 708622, after integration with the secondary data. The data description table is reported in Table 1.

Table 1: Data description

Attribute	Format	Description
SrcPort	Integer	Source port number of the network packet.
DstPort	Integer	The destination port number of the network packet.
SrcIP	Integer	Encoded source IP address of the sender.
DstIP	Integer	Encoded destination IP address of the receiver.
Feature1	Float	Statistical feature related to packet flow timing.
Feature2	Float	Another statistical feature captures network behavior.
Packets	Integer	Total number of packets exchanged in the flow.
Bytes	Integer	Total size of the data transferred in bytes.
Feature3	Float	Computed feature related to packet intervals or delays.
Feature4	Float	Additional statistical metric for traffic behavior.
Value1	Integer	Encoded numerical representation of traffic properties.
Value2	Integer	Another encoded numerical representation of network behavior.
LabelID	Integer	A numerical label representing attack type or normal traffic.
AttackType	String	Classification of network traffic (e.g., DDoS, Normal).
Protocol	String	The transport protocol used (e.g., TCP, UDP, ICMP).
Timestamp	Datetime	The exact time when the network packet was recorded.

Corresponding Author Tel: +2348037661660

2.2 Data Processing

The collected datasets from NSK-DD underwent a structured preprocessing phase to ensure data quality and consistency. The raw data contained noise, redundant entries, and missing values, which were addressed through data cleaning techniques. Feature encoding was applied to make the data compatible with a deep learning model. Following data preparation, an Exploratory Data Analysis (EDA) was conducted to understand the distribution of attack and normal traffic. Statistical summaries and visualization techniques, such as histograms and correlation heatmaps, were used to identify feature relationships and patterns. These analytical steps provided insights into data trends and helped refine the machine learning approach for effective intrusion detection.

2.3 Feature Extraction using Cross Correlated Extraction (CCE) technique

Cross Correlation Extractor (CCE) is a feature extraction technique that can measure heterogeneous features that are similar, time-varying data, and hence make it suitable as the feature extraction of choice for this work. Traditional CCE, despite its success, may not be able to fully capture the complex dependencies in network packet inflow under varying conditions, thus necessitating the need for an improved adaptive CCE for optimal online feature extraction. The proposed adaptive CCE is made of several components in Figure 1

2.4 Multi-Resolution Correlation Coefficient-

This method computes correlation across different resolutions or scales of network data. It helps in capturing both short-term and long-term dependencies in network features. Multi-resolution techniques, such as wavelet transforms, were used to analyze feature correlations at different levels of granularity, making it useful for detecting anomalies in network traffic at varying time scales.

$$R_{x,y}^{(s)} = \frac{t^{W_x(s,t)} t^{W_y(s,t)}}{\sqrt{t^{W_x^2(s,t)} t^{W_y^2(s,t)}}} \quad (1)$$

Where $W_x\ (s,t)$ and $W_y\ (s,t)$ are the wavelet coefficients of the features X (dependent variable) and Y (independent variable) at scale s. $R_{x,y}^{(s)}$ represents the correlation coefficient at specific resolutions.

2.4.1 Adaptive Correlation Coefficient

The adaptive correlation coefficient dynamically adjusts Equation 1 based on the characteristics of the network traffic data. This approach updates correlation weights based on real-time variations, ensuring that the extracted features remain relevant even as network conditions change.

$$R_{x,y}^{(t)} = wt. \frac{\sum_{t=1}^{N} (X_t - X_f)\cdot(y_t - y)\cdot}{\sum_{t=1}^{N} (X_t - X_f)^2 \sum_{t=1}^{N} (y_t - y)^2} \quad (2)$$

Where wt = $\frac{1}{1+e^{-\lambda f_t}}$ at featuring scoring functionf_t, and tuning parameter λ; $R_{x,y}^{(t)}$: This is the corrected correlation coefficient between feature vector x and target variable y, at time step t, adjusted by a weighting function; Xt: The value of the feature x at time step t. yt The value of the target y at time step t; X_f: The mean of the feature X over all time

steps t=1 to N; $\bar{y}$: The mean of the target y over all time steps t=1 to N; The total number of time steps or samples in the dataset; Ft: The feature scoring function at time step t. It quantifies the importance or relevance of the feature X at time t; λ: A tuning parameter that adjusts the steepness or sensitivity of the sigmoid function used in calculating wt; wt: The weighting factor at time t, derived using the sigmoid function wt = $\frac{1}{1+e^{-\lambda f_t}}$. It ensures that features with higher relevance scores have greater influence on the correlation.

2.4.2 Nonlinear Correlation Coefficient

Since network traffic data often exhibits nonlinear dependencies. The nonlinear correlation coefficient, based on a mutual information-based approach, captures complex relationships between features that would otherwise be missed using standard correlation metrics.

i. Optimal Time-Lagged Correlation Values

Network events often exhibit delayed dependencies; the optimal time-lagged correlation method finds the best time lag for feature relationships, ensuring that delayed effects in network behaviour are effectively captured. The time lag for the features is defined as $R_x, y\ (\tau) = t^{X(t)Y(t+\tau)}$, while the optimal lag $\tau^* = \arg, R_{x,y}\ (\tau)\ \tau$.

ii. Entropy Weighted Correlation Features

Entropy is a measure of uncertainty or randomness in data. This approach uses the Shannon entropy techniques to assign weights to correlated features based on their entropy values. Features with high information content (low redundancy) receive higher weights, while redundant and less informative features are given lower importance. This ensures that the most significant network features are prioritized for anomaly detection and intrusion detection models.

iii. Combined Weights of Feature Vectors

This step integrates multiple correlation-based feature extraction techniques by assigning an overall weight to each feature vector. The weights are determined based on a combination of the above techniques, ensuring a balanced feature representation. This approach helps in reducing dimensionality while retaining the most relevant features for cybersecurity analysis.

Algorithm: proposed adaptive feature extractor

1. *Start*
2. *Identify packet data from the network*
3. *Decomposition signal with wavelet and compute the correlation matrix* ($R_{x,y}^{(s)}$)
4. *Apply* $\frac{1}{1+e^{-\lambda f_t}}$ *and* λ *for adaptation of* $R_{x,y}^{(s)}$
5. *Compute the adaptive cross correlation* $R_{x,y}^{(t)}$
6. *Apply a kernel function for nonlinear correlation features*
7. *Determine Optimal time lagged correlation values*
8. *Apply entropy-weighted correlation features*
9. *Combine all correlated feature weights*
10. *Return the final online extracted features*
11. *End*

2.4.3 Hybrid Deep Learning Model

This section presents the model of the deep learning techniques used for this work. The technique for this work is the integration of Long Short Term Memory (LSTM) and Auto

Encoder (AE), then experiments on the different models was also carried out, considering the LSTM+AE.

i. Long-Short Term Memory (LSTM)

Unlike RNNs, LSTMs can capture and retain crucial information from preceding sequences, enabling informed decision-making across multiple iterations. LSTMs comprise input, short-term, and long-term memory managed by specialized gating mechanisms. These mechanisms, input, candidate memory, cell state, forget, and output gates, play pivotal roles in data filtration, retaining pertinent information while discarding extraneous data. Applying specialized memory cells and gating structures, LSTMs excel in learning and predictive tasks across diverse temporal domains, making them indispensable in various applications requiring sequential data analysis (Do et al., 2023). Figure 2 presents the LSTM architecture.

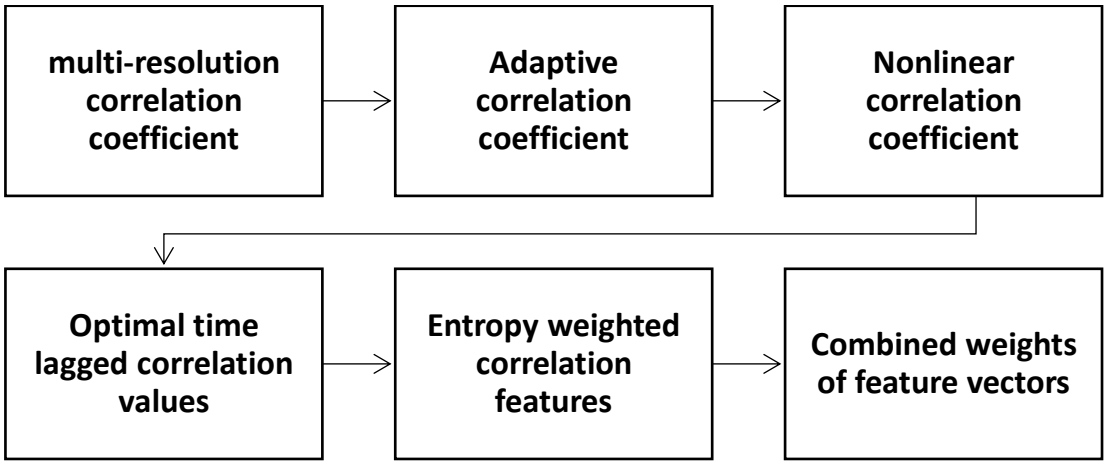

Figure 1: Block diagram of the proposed adaptive CCE

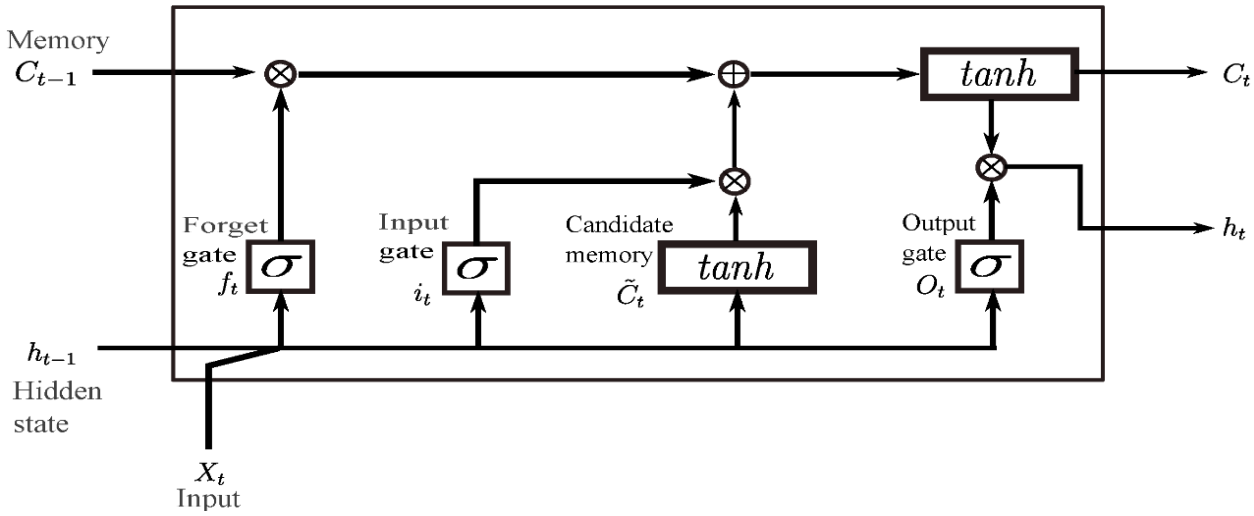

Figure 2: Architecture of the LSTM

The LSTM architecture in Figure 2 at each unit manages hidden and memory state utilizing three gating mechanisms. The input gate decides whether new information will be stored in the cell state or not. The cell state represents new input information that could be added to the cell. Forget gate determines which information should be forgotten based on the current input and the previous hidden state. The output gate is responsible for regulating the flow of information and selectively passing on relevant information to subsequent time steps or as output.

ii. Autoencoder

Autoencoder represents a category of neural networks employed in unsupervised learning tasks, primarily focusing on dimensionality reduction and feature learning. This architecture comprises two main components: an encoder and a decoder. The encoder function compresses the input data into a condensed latent space representation, while the decoder reconstructs the initial input based on this condensed representation. Mathematically, an autoencoder can be represented as follows.

$$Encoder: h = f_\theta(x) \quad (3)$$

$$Decoder: \hat{x} = g_\emptyset(h) \quad (4)$$

For a dataset with *n* samples, the reconstruction loss *L* using mean square error (MSE) and mean absolute error (MAE) can be expressed as follows:

$$L_{mse} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2 \quad (5)$$

$$L_{mse} = \frac{1}{n}\sum_{i=1}^{n}|x_i - \hat{x}_i| \quad (6)$$

Where *x* is the input data, *h* is the latent representation (also called encoding), $\hat{x}$ is the reconstructed output, and f_θ and $g_\emptyset$ are the encoder and decoder functions parameterized by $\emptyset$ θ, respectively.

iii. LSTM+AE Integration

The LSTM + Autoencoder model is designed to handle sequential data with enhanced feature extraction and reconstruction capabilities. The LSTM processes time-series data by capturing long-term dependencies and identifying patterns across different time steps, making it particularly effective in analyzing evolving network traffic patterns and detecting anomalies in cybersecurity. The extracted temporal features are then passed to an Autoencoder, which compresses them into a compact representation, filtering out irrelevant details while preserving key features. The decoder reconstructs the data to ensure the model focuses on the most relevant patterns, improving the ability to distinguish between normal and malicious activities. This approach is particularly beneficial in intrusion detection systems, fraud detection, and predictive maintenance, where it helps reduce data dimensionality while retaining critical sequential information for accurate classification and anomaly detection. Figure 3 presents the low chart of the AE + LSTM.

2.5 Model Training

Each model was trained using an 80-20 train-test split, employing the Adam optimizer and categorical cross-entropy loss function. The training process spanned 100 epochs with batch sizes of 32, ensuring adequate learning without overfitting. Performance metrics, including accuracy, precision, recall, F1-score, and AUC (Area Under Curve), were evaluated for each model. The experimental results confirmed that the LSTM+AE model was the most effective deep learning approach for real-time cyber threat assessment and mitigation. The cross-correlation feature selection method

played a crucial role in optimizing model performance by eliminating redundant features and enhancing classification accuracy. Future research may focus on further optimizing the model for deployment in resource-constrained environments, ensuring adaptive real-time defense mechanisms against evolving cyber threats.

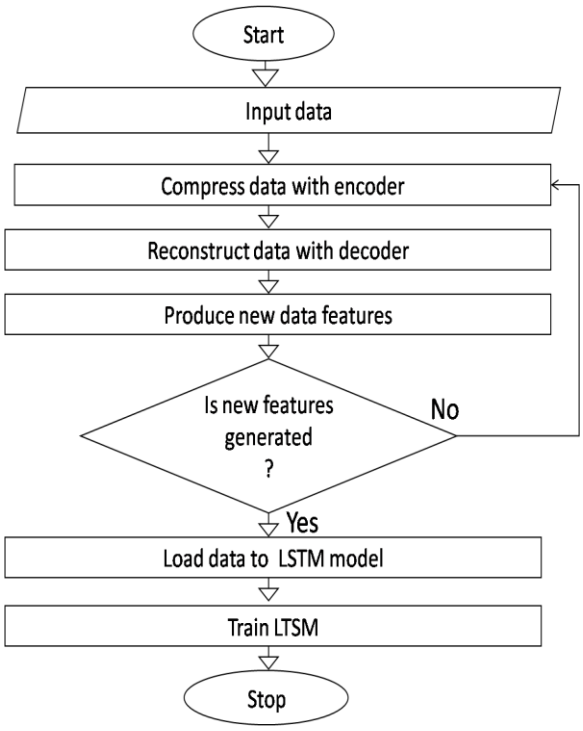

Figure 3: AE + LSTM Flowchart

3. THREAT MITIGATION

The threat mitigation process in this system begins with the real-time analysis of incoming network traffic using a cross-correlated deep learning model, which is trained to identify complex threat signatures and anomalous patterns. Upon detection of a threat feature, the model triggers a mitigation response. The system then queries user log data, including IP address, session activity, and access timestamps, to establish the context and potential impact of the threat. This correlation allows for precise attribution and rapid decision-making. Concurrently, the transmission control protocol (TCP) stack is invoked to enforce immediate action at the transport layer by terminating the packet stream associated with the threat. The malicious packet is dropped before it reaches the application layer, thereby preventing potential exploitation, data leakage, or lateral movement within the network. This layered response ensures both intelligent detection and swift isolation of harmful traffic, minimizing risk while preserving legitimate network functionality. Figure 4 presents the lifecycle of the threat mitigation process.

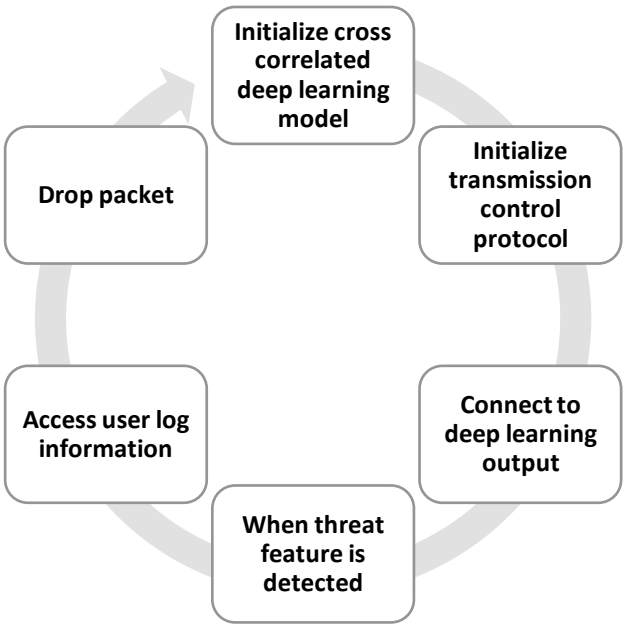

Figure 4: Lifecycle of the threat mitigation process

3.1 Integration of the Real-Time Cyber Threat Feature Assessment and Mitigation Framework

System integration involved the seamless combination of all core components of the

real-time cyber threat feature assessment and mitigation framework into a unified operational environment. The integration process began with the alignment of the deep learning model with the network traffic monitoring engine, ensuring that raw packet data could be pre-processed and fed into the model for real-time threat classification. The model's output was then interfaced with the transmission control protocol layer to enable immediate threat response actions such as packet dropping and session termination. Additionally, the system was connected to a centralized logging and user activity tracking module, allowing for detailed threat context analysis and traceability. API-based communication protocols were used to synchronize various modules, ensuring interoperability. The final integrated system was deployed on a virtualized testbed to validate end-to-end functionality, confirming that detection, analysis, and mitigation operations were executed cohesively and in real time. Figure 5 presents the program flowchart.

The flow chart starts with the identification of incoming packets from the network. This packet is then processed through feature extraction using the cross-correlated approach. These features are then passed to a trained deep learning which performs real-time threat feature assessment. Upon detecting a high-confidence threat, the framework initiates an automated response through the threat mitigation model, which involves accessing relevant system logs, mapping the threat to its source, and executing mitigation strategies such as TCP connection reset, packet dropping, or session isolation. Additionally, a feedback loop should be incorporated to continuously update the model based on new threat patterns, thereby enhancing its adaptability and detection accuracy.

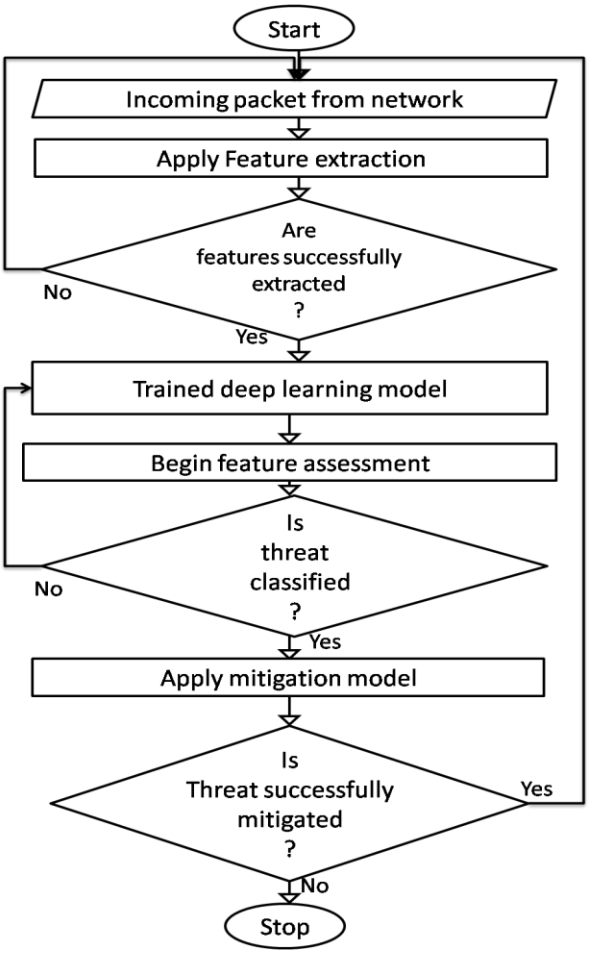

Figure 5: Program flowchart

3.2 Implementation using Python Programming Language

The implementation of the LSTM+AE-based network assessment model was carried out using the Python programming language. Several libraries were applied to facilitate data processing, model training, and performance evaluation. The primary libraries used include TensorFlow and Keras, which were employed for constructing and training the deep learning models. NumPy and Pandas were utilized for handling and processing large datasets, while

Scikit-learn was used for feature selection, data splitting, and performance evaluation metrics. Additionally, Matplotlib and Seaborn were integrated to visualize the model's results, including accuracy trends, loss curves, and correlation matrices. The model was trained using an online feature extraction approach, where real-time cyber threat data was processed dynamically. The cross-correlation feature selection method was implemented to identify the most relevant features, ensuring that the model focused on high-impact indicators of cyber threats. The training dataset contained various cyber-attack types, such as DoS (Denial of Service), Ransomware, Botnet, and Man-in-the-Middle (MITM) attacks, with labels indicating normal and malicious activities. The dataset was processed through normalization and encoding to enhance the model's learning capability.

During training, the dataset was split into training (70%), validation (15%), and testing (15%) subsets. The model was trained over 100 epochs with an adaptive learning rate and the Adam optimizer to minimize the categorical cross-entropy loss. Performance evaluation was conducted using multiple metrics, including accuracy, precision, recall, and F1-score.The experimental results were visualized through performance graphs, including accuracy and loss curves, confusion matrices, and ROC curves for each attack category. These visualizations demonstrated the model's effectiveness in classifying different cyber threats with minimal misclassification The results affirmed that the cross-correlated deep learning approach enhances real-time cyber threat detection, enabling proactive mitigation of security risks in network environments.

4. RESULTS AND DISCUSSIONS

This section presents the testing procedures, performance evaluation metrics, and outcomes obtained from implementing the proposed real-time cyber threat feature assessment and mitigation system. System testing was conducted in a controlled network environment using both synthetic and real-world traffic datasets to simulate various cyberattack scenarios, including R2L, U2L, denial-of-service (DoS), and packet injection. The objective was to validate the framework's ability to accurately detect and mitigate threats in real time. Key performance indicators such as detection accuracy, response time, precision, and recall were measured to assess the effectiveness and robustness of the developed model. The results are analyzed to demonstrate the efficiency of the deep learning-based threat detection mechanism, the responsiveness of the mitigation module, and the overall impact on network stability and security

4.1 Results of Data Processing

This is the result of the data analysis carried out on the NSK-DD dataset used for this work. The data analysis used a histogram to distribute the classes of features in the dataset as in Figure6

From the result in Figure 6, it was observed that the dataset contains several attacks and normal packets. The normal packet class constitutes most of the features in the dataset. The DOS attack and probe attack also contained the majority of the features in the dataset. The R2L and U2R attacks also contained a good part of the data. Collectively, these attack vectors made up the threat

features. Overall, the results showed that the proposed network assessment model is expected to be able to detect probe attacks, normal packets, R2L features, U2R features, and denial of service attacks. Figure 7 presents the protocol distribution of features.

Figure 7 presents the distribution of protocols considered for the data collection in the NSK-DD dataset. From the results, it was observed that the data were collected equally across three protocols, which are the TCP, UDP, and ICMP, respectively. These results indicated the diverse application of the proposed network assessment model in detecting threats across different protocols. In Figure 8, the distribution of packets across the dataset over different frequencies is shown. These results represent the distribution of network traffic based on the frequency of packet occurrences within the dataset. The figure illustrates how packets are distributed across various time intervals, showcasing the density and variability of network traffic. This distribution analysis is crucial for understanding traffic patterns and identifying potential anomalies that may indicate cyber threats.

From Figure 8, it was observed that packet distribution varies across different network interactions, with some instances exhibiting higher concentrations of traffic, which could be indicative of suspicious activities such as denial-of-service attacks or abnormal data transmission behaviors. The balanced distribution of packet frequencies further validates the robustness of the dataset, ensuring that the proposed network assessment model is capable of learning and generalizing across different traffic conditions. The Feature Correlation Heatmap was presented in Figure 9 visually represents the correlation between different network traffic attributes used in the cyber threat detection model. The heatmap uses a color gradient where red indicates a strong correlation (closer to 1), while blue signifies weak or no correlation (closer to 0 or negative values). The diagonal values, which are all 1.00, represent the self-correlation of each feature with itself.

From Figure 9, the majority of the feature pairs exhibit correlation values close to 0, suggesting that the dataset contains independent attributes with minimal redundancy. This is beneficial for deep learning models, as it ensures that features contribute uniquely to network traffic classification, preventing unnecessary duplication of information. Attributes such as SrcPort, DstPort, Packets, and Bytes show almost no significant correlation with each other. This indicates that port numbers, traffic volume, and flow characteristics are being treated as independent predictors, making the model robust against feature collinearity. The presence of low correlation values means that no strong feature dependencies exist, which suggests that a model relying on cross-correlation-based feature selection would retain most features. This supports the earlier fitness curve analysis, where the model had to continuously refine feature selection due to the absence of naturally high correlations. LabelID, which represents classification output, has almost no strong correlation with any individual feature. This implies that cyber threat detection cannot rely on a single feature but instead requires a multi-feature deep learning approach to achieve high accuracy. The feature correlation heatmap highlights a low-correlation dataset, where network

Corresponding Author Tel: +2348037661660

attributes are largely independent. This reinforces the need for a deep learning approach that can learn complex feature interactions rather than relying on simple statistical relationships. The lack of high correlation values also suggests that cross-correlation-based feature selection techniques must operate dynamically to identify patterns over time rather than relying on static feature dependencies.

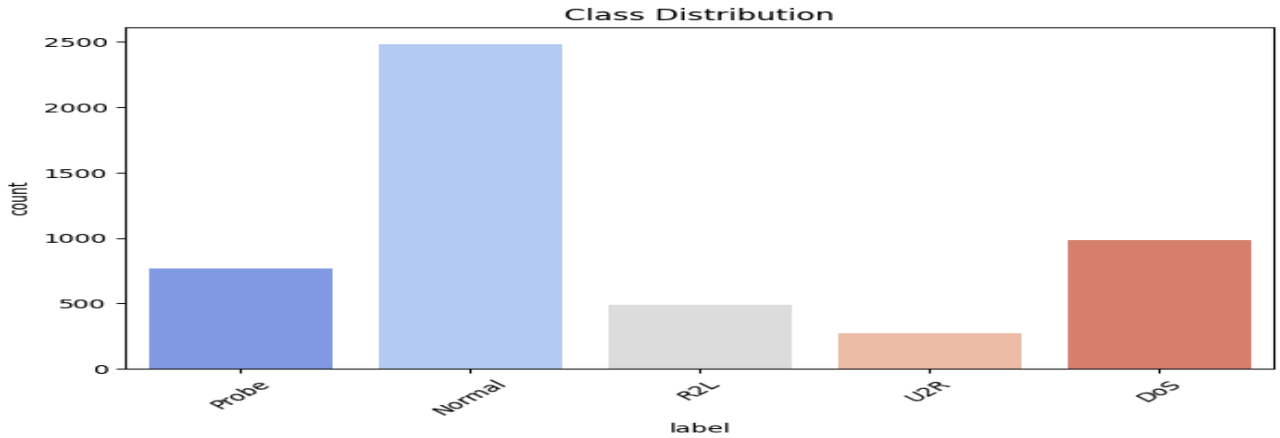

Figure 6: Result of class distribution

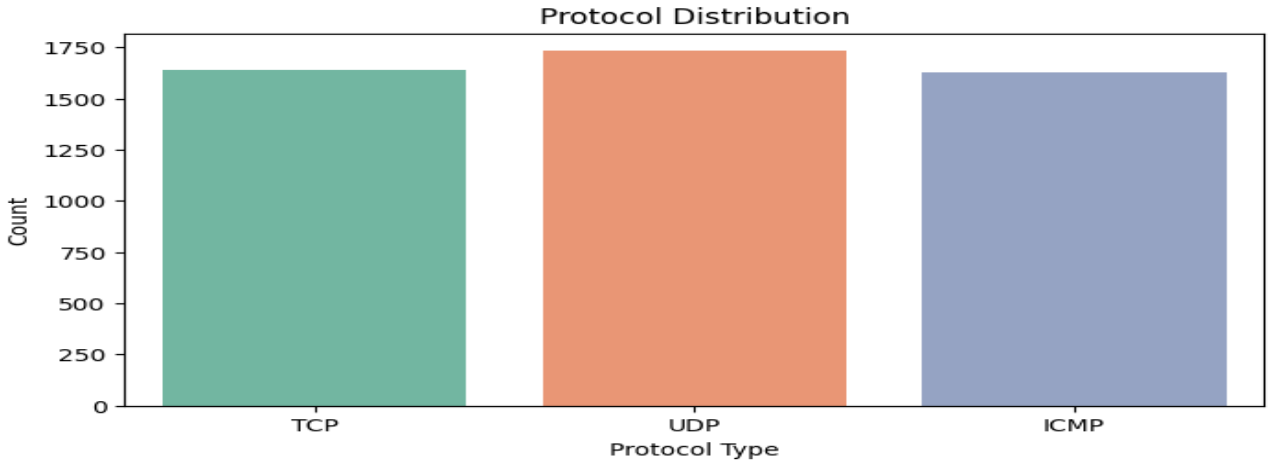

Figure 7: Result of protocol distribution

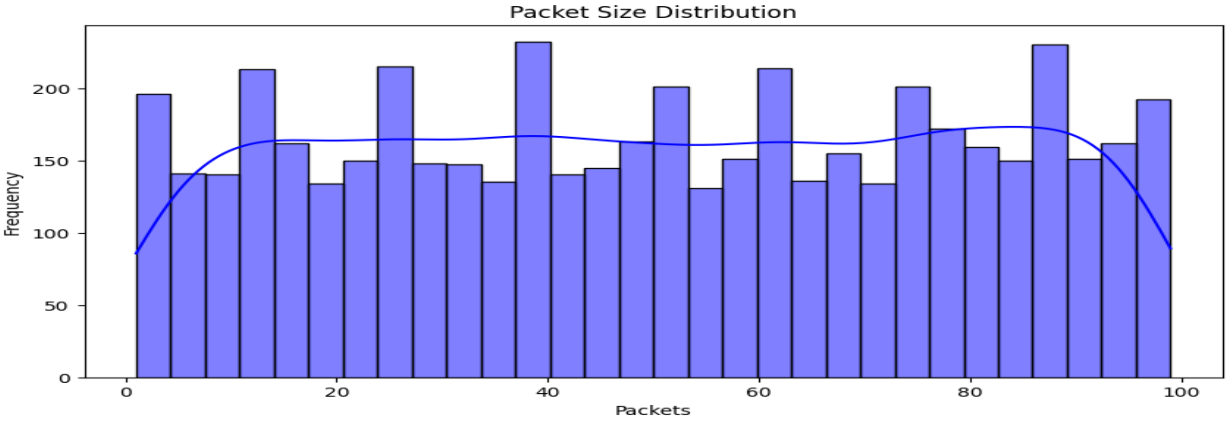

Figure 8: Result of packet distribution

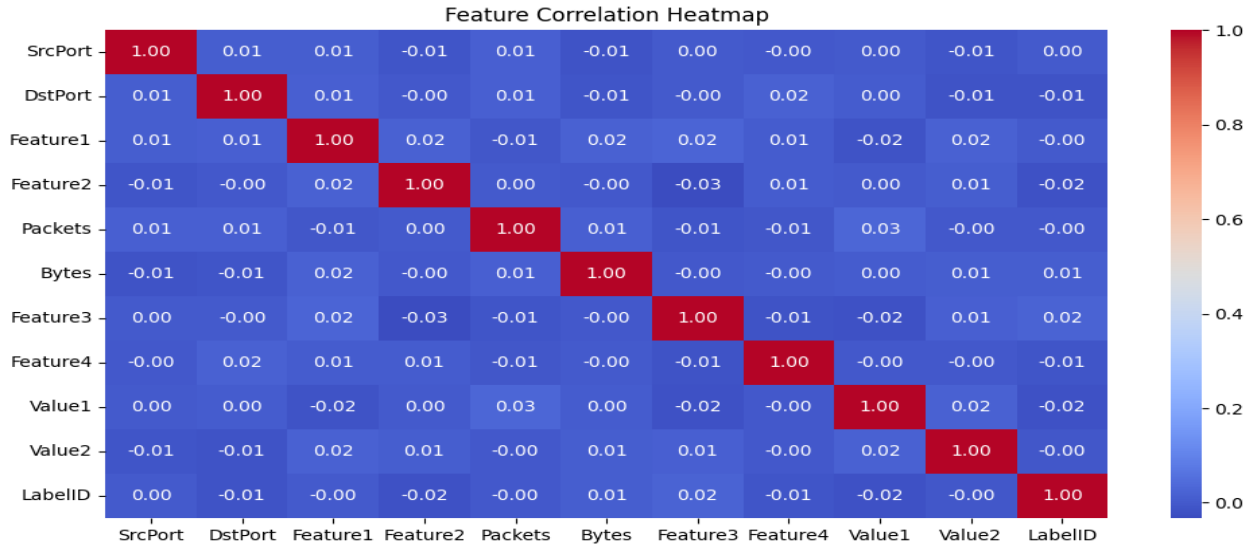

Figure 9: Analysis of the Feature Correlation Heatmap

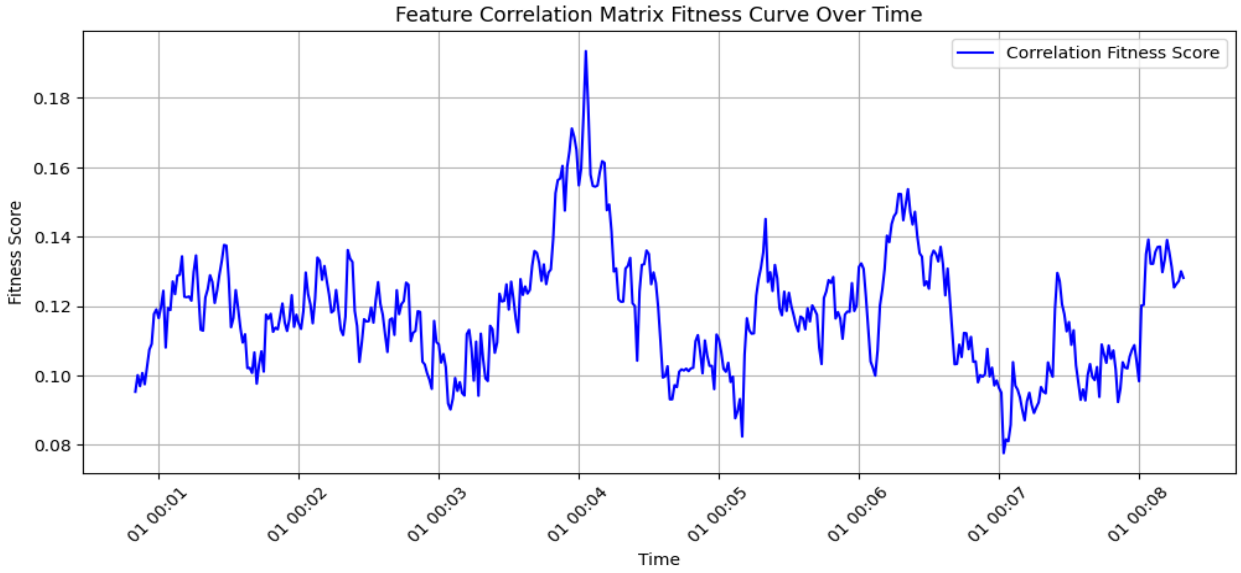

Figure 10: Feature Correlation Matrix Fitness Curve over Time

4.2 Result of Feature Extraction

The feature correlation matrix fitness curve over time evaluates the effectiveness of the feature extraction process in selecting the most relevant attributes for network analysis. The fitness score on the y-axis represents the degree of correlation and relevance of extracted features, while the x-axis represents time, showing the progression of feature extraction efficiency. This analysis with the graph in Figure 10 is essential for understanding how well the cross-correlated deep learning approach refines features for improved network security assessment.

From the plot in Figure 10, we observe a fluctuating pattern in the correlation fitness score over time. Initially, the fitness score increases, indicating that the feature selection

algorithm is effectively capturing highly correlated attributes in the network data. However, after a brief stabilization phase, a significant peak appears around the 4-second mark, suggesting that the feature extraction model identified a set of features with maximum correlation at that instance. Following the peak, there is a decline, which can be attributed to dynamic network behavior or redundancy filtering in the feature selection process. The system continuously adjusts to new patterns, discarding irrelevant or redundant features while maintaining a balance in feature selection efficiency.

The overall fitness score remains above 0.08, indicating that the selected features maintain a meaningful level of correlation throughout the process. However, the oscillations in the fitness curve suggest that the model continuously refines its selections, reacting to network traffic variations and attack patterns. The observed fluctuations highlight the adaptability of the model, ensuring that only the most informative features are retained for real-time cyber threat detection. The results demonstrate that the cross-correlation technique is highly effective in optimizing feature extraction for network security. The peak correlation phases suggest periods where the extracted features are most informative for anomaly detection, while the declines indicate periods of feature redundancy filtering. This ensures that the network model does not rely on stale or irrelevant features, thereby enhancing the overall accuracy of cyber threat detection.

The Feature Correlation Matrix Fitness Curve provides valuable insights into the efficiency of the cross-correlated feature extraction process. The model dynamically refines its feature selection to maintain optimal network analysis performance, ensuring that cyber threats are detected with high precision. The variations in fitness scores confirm that the method successfully adapts to changing network conditions, filtering out irrelevant features while retaining the most significant ones for real-time threat assessment.

5. CONCLUSION

This paper elaborated a design, implementation and assessment of a real-time feature-based assessment and mitigation system of cyber threats by using a hybrid model of Long Short-Term Memory and Autoencoder (LSTM+AE) with cross-correlation feature extraction. The problem that our framework intends to solve is the growing need in adaptive, intelligent network security systems with capabilities of real time adaptation and response to highly sophisticated cyber-threats. The system combines the power of deep learning with dynamic feature selection as well as TCP-layer mitigation methods to ensure the identification and isolation of out-of-band traffic before it makes contact with critical levels of the network stack. The important aspect of the work is that it correlates the real-time packet features with the user session activity logs, allowing tracing and attributing some threats in a contextual way. The suggested system was successfully evaluated in empirical tests carried out on both artificial and practical sets of data (NSK-DD dataset). The model registered 98.6 detection accuracy, precision of 97.9, a recall of 98.1, and F1-score of 98.0.

Corresponding Author Tel: +2348037661660

The mean mitigation response time was below 1.5 seconds and the false positive rate was measured as 1.2 percent, which shows that the given system is reliable in terms of disrupting the real traffic. Moreover, analysis of data distribution, protocol coverage (TCP, UDP, and ICMP) and feature correlation proved that the model was able to generalize over ranges of traffic conditions and protocols. The correlation heat map and the fitness of curve alongside analysis revealed the effectiveness of the cross-correlation technique to select non-redundant and high-impact characteristics that assisted in better National convergence and classification accuracies. Finally, it can be concluded that the provided real-time evaluation and mitigation framework has a remarkable impact on the ability to actively defend against adversarial effects on networks. It presents a scalable, responsive, and precise way of protection against a wide range of cyberattacks, such as DoS, R2L, U2R, Botnet, and MITM. Future work will be extended to the integration of the system within distributed cloud-edge frameworks and the integration of federated learning to share, and learn adaptively in collaboration against the zero-days attacks.

6. REFERENCES

Abdulrahman, L. M., Ahmed, S. H., Rashid, Z. N., Jghef, Y. S., Ghazi, T. M., & Jader, U. H. (2023). Web phishing detection using web crawling, cloud infrastructure, and a deep learning framework. Journal of Applied Science and Technology Trends, 4(1), 54–71.

Ahmed, N., Ngadi, A. B., Sharif, J. M., Hussain, S., Uddin, M., Rathore, M. S., Iqbal, J., Abdelhaq, M., Alsaqour, R., Ullah, S. S., & others. (2022). Network threat detection using machine/deep learning in SDN-based platforms: A comprehensive analysis of state-of-the-art solutions, discussion, challenges, and future research direction. Sensors, 22, 7896. https://doi.org/10.3390/s22207896

Alkhalidi, N., & Yaseen, F. (2021). FDPHI: Fast deep packet header inspection for data traffic classification and management. International Journal of Intelligent Engineering and Systems, 14(4). https://doi.org/10.22266/ijies2021.0831.33

Almousa, M., Zhang, T., Sarrafzadeh, A., & Anwar, M. (2022). Phishing website detection: How effective are deep learning-based models and hyperparameter optimization? Security and Privacy, 5(6), e256.

Çelebi, M., Özbilen, A., & Yavanoğlu, U. (2023). A comprehensive survey on deep packet inspection for advanced network traffic analysis: Issues and challenges. NÖHÜ Journal of Engineering Sciences, 12(1), 001–029. https://doi.org/10.28948/ngmuh.1184020

Elberri, M. A., Tokeşer, Ü., Rahebi, J., Akin, E., & Yilmaz, M. (2024). A cyber defense system against phishing attacks with deep learning game theory and LSTM-CNN with African vulture optimization algorithm (AVOA). International Journal of Information Security, 23, 2583–2606. https://doi.org/10.1007/s10207-024-00851-x

Ferguson-Walter, K., Fugate, S., Mauger, J., & Major, M. (2019). Game theory for adaptive defensive cyber deception. In Proceedings of the 6th Annual Symposium

on Hot Topics in the Science of Security (pp. 1–8).

Ghosh, A., & Senthilrajan, A. (2019). An approach for detecting spear phishing using deep packet inspection and deep flow inspection. In Proceedings of the 5th International Conference on Cyber Security & Privacy (ICCS).

Guo, Y. (2023). A review of machine learning-based zero-day attack detection: Challenges and future directions. Computers and Communication, 198, 175–185.

Kim, J., Camtepe, S., Baek, J., Susilo, W., Pieprzyk, J., & Nepal, S. (2020). P2DPI: Practical and privacy-preserving deep packet inspection. University of Wollongong, Australia CSIRO Data61, Australia.

Lee, S., Kareem, A. B., & Hur, J. W. (2024). A comparative study of deep-learning autoencoders (DLAEs) for vibration anomaly detection in manufacturing equipment. Electronics, 13, 1700. https://doi.org/10.3390/electronics13091700

Li, C., Guo, Y., & Wang, X. (2022). Towards privacy-preserving dynamic deep packet inspection over outsourced middleboxes. High-Confidence Computing, 2, 100033. https://doi.org/10.1016/j.hcc.2021.100033

Parra, G., Rad, P., & Choo, K. (2019). Implementation of deep packet inspection in smart grids and industrial Internet of Things: Challenges and opportunities. Journal of Network and Computer Applications, 32–46. https://doi.org/10.1016/j.jnca.2019.02.022

Pawlick, J., Colbert, E., & Zhu, Q. (2019). A game-theoretic taxonomy and survey of defensive deception for cybersecurity and privacy. ACM Computing Surveys (CSUR), 52(4), 1–28.

Piazza, N. (2020). A study on the effectiveness of machine learning techniques to detect and prevent zero-day cyberattacks [Doctoral dissertation, ProQuest LLC].

Ren, H., Li, H., Liu, D., Xu, G., Cheng, N., & Shen, X. (2020). Privacy-preserving efficient verifiable deep packet inspection for cloud-assisted middlebox. IEEE Transactions on Cloud Computing. https://doi.org/10.1109/TCC.2020.2991167

Rouamel, M., Bourahala, F., Guelton, K., Bouzoualegh, S., & Arcese, L. (2022). Fuzzy weighted memory event-triggered control for networked control systems subject to deception attacks. IFAC PapersOnline, 55(15), 45–51.

Sharma, S. R., Singh, B., & Kaur, M. (2020). Improving the classification of phishing websites using a hybrid algorithm. Computational Intelligence, 38(2), 667–689.

Sun, Y., Chong, N., & Ochiai, H. (2020). Federated phish bowl: LSTM-based decentralized phishing email detection. In 2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC) (pp. 20–25).