Federated Learning-Based Ensemble CNN for Network Intrusion Detection in Healthcare Systems

Dr. T. Revathi 1, Mr. R. Prabhu 2, Mr. K. Vijesh Pethuram 3  
Department of Information Technology, Mepco Schlenk Engineering College

***Abstract*— The rising complexity of cyber threats in Industry 5.0-driven healthcare environments necessitates robust, privacy-preserving intrusion detection mechanisms. In this paper, we propose a Federated Learning-Based Ensemble Convolutional Neural Network (FLE-CNN) for Network Intrusion Detection in smart healthcare systems. Unlike traditional centralized models, our approach leverages federated learning to preserve data privacy by enabling collaborative model training across multiple decentralized healthcare systems without exposing sensitive patient data. The FLE-CNN integrates multiple models like CNN, SVM, DT and KNN into an ensemble, enhancing the detection accuracy of diverse intrusion types while mitigating overfitting. We utilize the NSL-KDD dataset to assess the model's performance. Experimental results demonstrate that the proposed FLE-CNN framework outperforms existing intrusion detection schemes in terms of accuracy, recall, precision, and F1-score, while also addressing key security concerns related to data sharing and system scalability. This work presents a significant advancement toward secure, intelligent, and privacy-preserving intrusion detection in next-generation healthcare infrastructures.**

***Index Terms*— Federated Learning, Network Intrusion Detection, Smart Healthcare, Convolutional Neural Network, Data Privacy.**

# I. INTRODUCTION

The evolution toward Industry 5.0 has revolutionized healthcare delivery through personalized and intelligent services powered by the Internet of Medical Things (IoMT), Artificial Intelligence (AI), and real-time data analytics. However, the increased connectivity of medical systems has exposed them to numerous cybersecurity vulnerabilities. Network Intrusion Detection Systems (NIDS) play a crucial role in safeguarding these systems, particularly when sensitive patient data is constantly transmitted across networks. Traditional centralized NIDS approaches face challenges such as data privacy risks, communication overhead, and vulnerability to single points of failure.

To overcome these limitations, federated learning (FL) has emerged as a decentralized paradigm that facilitates collaborative model training without compromising data privacy. In this work, we propose an intrusion detection scheme that leverages an ensemble of convolutional neural network (CNN), Support Vector Machine (SVM), K-Nearest Neighbours (KNN) and Decision Tree (DT) trained within a federated learning framework. This novel approach ensures robust and generalized intrusion detection performance across heterogeneous and distributed healthcare systems while addressing the stringent privacy regulations prevalent in the medical domain.

Additionally, our model incorporates four key modules—Capture Engine, Cloud Server, Proxy Server, and Hospital Server—that work together to identify and respond to intrusions in real-time. The proxy server acts as a decoy to trap attackers and collect malicious patterns, while the capture engine continuously sniffs traffic for anomalies. Leveraging periodic federated training and secure model aggregation, FLE-CNN ensures faster predictions, reduced bandwidth use, and superior anomaly detection accuracy. This paper outlines the design, implementation, and evaluation of the proposed system using the NSL-KDD dataset, validated by performance metrics and formal security analysis using the Scyther tool. The results demonstrate the system’s ability to deliver high accuracy while preserving privacy, making it an effective solution for safeguarding Industry 5.0 healthcare infrastructures.

*A. Research Motivation*

The increasing adoption of smart healthcare technologies in Industry 5.0 environments has introduced new challenges in ensuring the security and privacy of sensitive medical data. Wearable sensors, remote diagnostics, and AI-driven patient monitoring systems have made healthcare delivery more efficient and personalized. However, these interconnected systems also present a significantly expanded attack surface for cybercriminals. Traditional intrusion detection systems (IDS) that rely on centralized data collection and analysis struggle to keep up with modern threat landscapes. These systems often require large volumes of patient data to be transmitted to a central location for model training, raising serious privacy concerns and increasing bandwidth consumption. Moreover, centralized systems are prone to single points of failure and may not effectively adapt to evolving attack vectors across diverse hospital networks.

Recent advances in federated learning offer a promising solution by enabling collaborative model training across distributed nodes without sharing raw data. When combined with ensemble-based deep learning techniques, particularly Convolutional Neural Network (CNN), this approach enhances detection accuracy, adaptability, and robustness. This leads the motivation for this research.

*B. Research Contributions:*

This work proposes a novel privacy-preserving intrusion detection framework for Industry 5.0 healthcare environments by integrating Federated Learning with Ensemble and Convolutional Neural Network (CNNs). Unlike traditional centralized models, our approach decentralizes the training process, enabling individual hospital nodes to collaboratively train an intrusion detection system (IDS) without sharing sensitive patient data. This ensures enhanced privacy compliance while also improving scalability and system robustness. The proposed FLE-CNN architecture incorporates an ensemble of CNNs, each trained with different configurations across nodes. The ensemble design improves detection accuracy, reduces overfitting, and enhances the generalizability of the system across diverse attack patterns. To support real-time intrusion detection, we introduce a modular architecture comprising a Capture Engine Module, a Cloud FL Server, a Proxy Server for decoy-based malware collection, and a Hospital Server acting as the primary data node. These modules work in unison to detect anomalies efficiently and provide immediate alerts. Also, we include a proxy server that acts as a honeypot to trap attackers. This proxy mimics real hospital servers and collects malware samples, which are then used to fine-tune the global detection model via federated learning. This approach not only enhances the system’s learning capability but also improves the anomaly detection rate in future iterations. The framework is evaluated using the NSL-KDD dataset, where it demonstrates high accuracy, precision, recall, and F1-score in detecting network intrusions. Additionally, to validate the security of the proposed design, we employ the Scyther tool for formal security analysis. This verification confirms the system’s resilience against multiple attack vectors such as man-in-the-middle attacks, DDOS, DOS, and port scans.

*C. Organization of the paper*

The paper's remaining sections are arranged as follows. The specifics of other comparable intrusion detection systems that are currently in use and relevant to smart healthcare are included in Section II. Section III lists the system design for the suggested FLE-CNN. Section IV provides the planned EIDS-HS's specifics. Additionally, Section V provides the EIDSHS's practical implementation. The specifics of several comparisons between the suggested FLE-CNN and other current methods are provided in Section VI. Section VII then presents the crucial security analysis of FLE-CNN. Additionally, Section VIII does the formal security verification utilizing the Scyther tool.

# II. Related Works

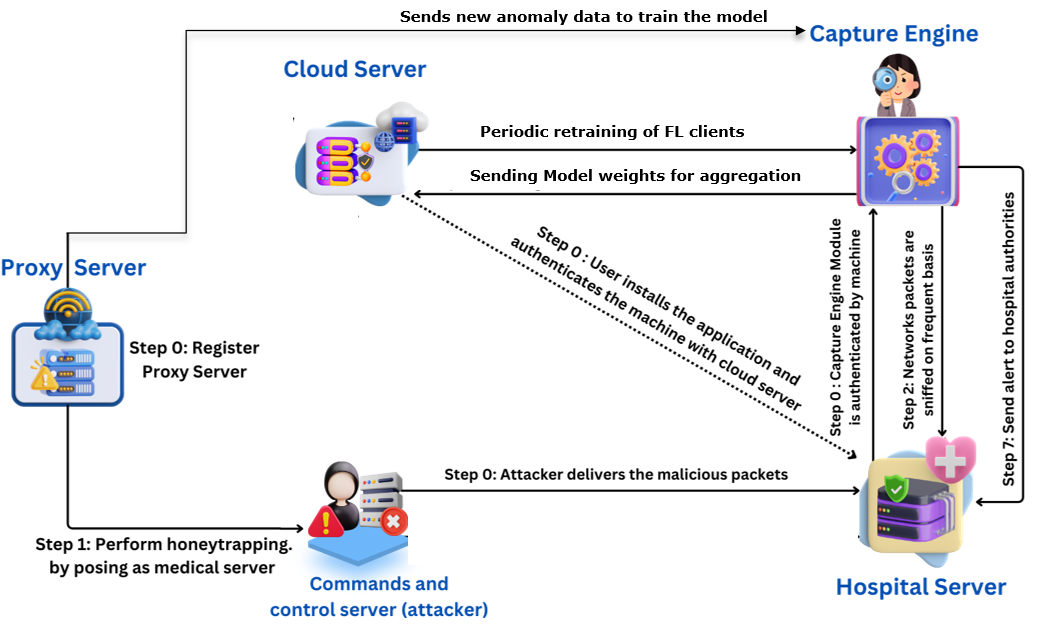
The integration of federated learning (FL) into intrusion detection systems (IDS) has garnered significant attention, particularly in the context of healthcare and IoT environments. A comprehensive survey by Belenguer et al. [1] categorizes FL-based IDS approaches, highlighting their potential in preserving data privacy while maintaining detection efficacy. Similarly, Agrawal et al. [2] discuss the challenges and future directions of FL in IDS, emphasizing the need for robust aggregation strategies and addressing issues related to data heterogeneity.

In the realm of healthcare, FIDChain [3] presents a blockchain-enabled federated learning (FL) framework specifically designed for IoT-based healthcare applications. The architecture combines the strengths of blockchain technology and federated learning to address critical challenges such as data privacy, traceability, and model integrity in distributed healthcare environments. By integrating blockchain, FIDChain ensures that all model updates and transactions are recorded in a tamper-proof ledger, thereby enhancing trust and accountability among participating nodes. The framework utilizes lightweight artificial neural networks (ANNs) to accommodate the limited computational resources of IoT healthcare devices, making it practical for real-world deployment in edge-based medical systems.

Another study by Almaghthawi et al. [4] introduces a blockchain-based federated learning intrusion detection system (FL-IDS) designed to enhance cybersecurity in distributed networks while preserving data privacy. In their approach, machine learning models are collaboratively trained across multiple nodes without the need to share raw data, addressing the critical privacy concerns associated with sensitive domains such as healthcare and industrial IoT. The framework employs blockchain technology to provide a decentralized, immutable ledger for tracking model updates and verifying the integrity of the contributions from each participating node.

For IoT networks, Nguyen and Beuran [5] propose FedMSE, a semi-supervised FL approach combining Shrink Autoencoder and Centroid one-class classifier to enhance anomaly detection in decentralized settings. Additionally, Gourceyraud et al. [6] explore an unsupervised FL-based IDS, introducing a federated K-means++ initialization technique to improve clustering performance without compromising data privacy.

In the context of the Internet of Healthcare Things (IoHT), a study by Almarashdeh et al. [7] proposes a privacy-preserving FL framework integrating differential privacy mechanisms to secure sensitive health data during collaborative model training. Their approach emphasizes minimizing the risk of data leakage during transmission by ensuring that only noise-perturbed model updates are shared among participating healthcare nodes. This not only safeguards the privacy of patients' medical records but also complies with regulatory standards like HIPAA and GDPR. The framework incorporates a lightweight deep learning model suitable for resource-constrained IoHT devices, enabling efficient real-time anomaly detection without compromising on performance. Additionally, the authors demonstrate how their system maintains high detection



accuracy while resisting membership inference and model inversion attacks, highlighting its practical applicability in real-world healthcare settings. This work serves as a foundational reference for integrating advanced privacy-preserving mechanisms into federated intrusion detection systems tailored for smart healthcare environments.

Fig. 1 Process flow of Proposed FLE-CNN IDS

Lazzarini et al. [8] evaluated the use of federated learning for intrusion detection in IoT environments, employing shallow artificial neural networks and various aggregation algorithms. Their study demonstrated that federated learning approaches could achieve comparable performance to centralized models while preserving data privacy. The research also highlighted the impact of different aggregation methods on model efficacy.

These studies collectively underscore the efficacy of FL in enhancing IDS capabilities across various domains, addressing challenges related to data privacy, system scalability, and detection accuracy.

# III. System Design

The Federated Learning-Based Ensemble Convolutional Neural Network (FLE-CNN) system shown in Fig.1 for intrusion detection in healthcare environments is structured around four main modules—Capture Engine Module, Cloud Server, Proxy Server, and Hospital Server—which together form a decentralized, privacy-aware, and intelligent intrusion detection framework. The Hospital Server Module acts as the central node within the hospital’s secure network. It authenticates communication with the cloud server, orchestrates local training of the ensemble which consists of   
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
SVM, KNN, DT and a CNN model, and handles data from IoT devices and internal systems. It also integrates the capture engine and facilitates real-time anomaly detection. The hospital server accumulates insights from its own traffic patterns, including those from ESP32 sensors, and prepares model updates for federated training. The Cloud Server coordinates the federated learning process. It collects encrypted model weights (not raw data) from multiple hospital servers and applies the Federated Averaging to generate global weights. These weights are then distributed back to each client (hospital node) for the next round of local training. The use of federated learning ensures that patient-sensitive data remains localized, preserving privacy and complying with healthcare data protection regulations.

A key enhancement to this system is the   
Proxy Server Module, which acts as a decoy designed to interact with potential attackers. By mimicking a legitimate medical server, it attracts malicious actors and captures attempted intrusions and malware payloads. These interactions are not only logged but also transformed into labelled anomalous data points that are used to enrich the global model’s knowledge base during future federated training cycles. At the edge of the network, the Capture Engine Module is deployed within the hospital's local infrastructure. It is responsible for continuously sniffing network traffic, including packets generated by connected healthcare IoT devices such as ESP32 boards with attached pulse sensors. These devices monitor patient vitals in real-time and communicate with the hospital server over Wi-Fi. The capture engine intercepts packets from such devices, extracts relevant features (e.g., protocol type, source/destination bytes, flag bits), and forwards them to the local ensemble CNN model for classification. The model then predicts whether each packet is benign or anomalous. Crucially, this process occurs without sending raw data outside the local network, aligning with the privacy goals of federated learning. Together, these modules form a cyclical and collaborative learning framework. Periodic retraining occurs using the newly captured anomalies and benign traffic, including data originating from medical IoT devices like the ESP32. This continuous improvement loop enhances the detection model’s accuracy and adaptability while maintaining low latency, reduced bandwidth usage, and high data privacy.

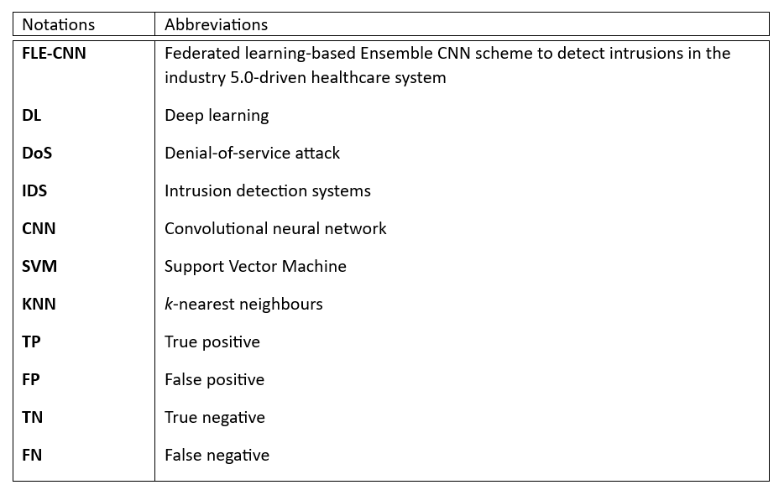
# IV. The Proposed Intrusion Detection Process

The proposed Federated Learning-based Ensemble CNN system in Fig.1 is designed to detect intrusions in a decentralized smart healthcare environment. The architecture involves interaction between the Cloud Server, Capture Engine, Hospital Server, Proxy Server, and the potential attacker’s command and control server. Each step of the process illustrated in the diagram corresponds to critical events in both normal operation and during a malicious attack attempt.

Initially, when the Capture Engine Module is installed on a local machine, the user must authenticate the device with the Cloud Server. This step ensures that only verified machines participate in the federated learning process. The authentication process may include device verification and secure registration to prevent rogue devices from joining the FL network. Simultaneously, a Proxy Server is registered within the system. This proxy mimics a legitimate medical server and is strategically deployed to act as a honeypot. It is used to lure attackers and gather malicious traffic without affecting actual hospital infrastructure. This module plays a crucial role in proactively detecting new or zero-day attacks. The Proxy Server actively pretends to be a vulnerable or legitimate hospital device. It interacts with attackers or malware sources by exposing dummy services or interfaces. When an attacker attempts to exploit this fake node, their activities are recorded in detail—including commands, payloads, and attempted breaches. The Capture Engine begins frequent and continuous sniffing of network packets within the hospital LAN/Wi-Fi. It captures data from internal systems as well as from connected medical IoT devices like ESP32 with pulse sensors. These packets are passed through a local ensemble model, which classifies them as either benign or anomalous.

At this point, an attacker unknowingly delivers malicious packets to the Proxy Server, thinking it is a real hospital machine. These packets are logged, analysed, and labelled as anomalous data, which is crucial for enhancing the detection model. These samples will later be incorporated into the FL training loop to improve model robustness. If the local model identifies a packet as anomalous (e.g., a DoS attempt, scanning, or malware delivery), an alert is triggered and sent to hospital authorities. This real-time alerting mechanism helps IT security teams respond quickly to prevent data breaches, system downtime, or patient safety risks. In parallel, each hospital node periodically sends its model weights to the Cloud Server. The cloud server aggregates these updates using Federated Averaging and sends back the improved global model. Importantly, no raw patient or network data is shared—only encrypted model weights—thus preserving privacy and complying with medical data regulations. Newly captured attack data from the Proxy Server and Hospital Server (including malicious packets delivered by attackers or suspicious traffic from IoT devices) is periodically used to retrain local models. This updated data helps the models evolve and stay resilient against new and evolving threats

# Table I Notations Used In Eids-Hs



The details of the notations used in the proposed EIDS-HS are given in Table I.

*A. Data Acquisition*

The proposed FLE-CNN intrusion detection system was evaluated using the NSL-KDD dataset [9], a refined and improved version of the widely used KDD CUP 99 dataset. This dataset includes a wide range of attack types such as denial-of-service (DoS), probing and various attacks. It contains records of approximately five million network connections, each characterized by 41 features. These features encompass various aspects of network behavior, such as protocol type, service, duration, and source/destination statistics, making the dataset well-suited for intrusion detection research. In the context of this study, a binary classification approach was adopted. The goal was to classify each record as either a normal connection or an intrusion. The dataset includes a total of 79,179 normal instances and 81,161 intrusion instances, providing a balanced and realistic testing ground for evaluating the performance of FLE-CNN.

*B. Preprocessing:*

To prepare the dataset for model training and evaluation, several preprocessing steps were undertaken. Initially, all duplicate records were removed to ensure quality of the dataset.

|  |
| --- |
| **Algorithm 1:** Federated Local Ensemble Model Training (FL Clients) |
| **Input**: Local dataset at each hospital **Parameters**: k for KNN, number of communication rounds R, Real-time packet stream P **Output**: Model weights:  ***Training Phase:***  **1. For each hospital client Hi ∈ H:** 2. Load the local dataset 3. Remove duplicate records 4. Split into feature matrix X and target variable y 5. Perform train-test split (70/30) 6. Apply standard scaling to numeric features 7. Apply one-hot encoding to categorical features in X 8. Apply label encoding to the target variable y 9. Apply PCA to reduce dimensionality 10. Train models: SVM, KNN (with k), DT, and CNN 11. Eval model accuracies: ​ 12. Compute normalized ensemble weights:  13. Send local weights ​ to the cloud server 14. **End For** |
| ***Real-Time Prediction Phase:***  15. **For each incoming packet p ∈ P:** 16. Apply preprocessing steps (scale, encode, PCA) to the real-time packet data. 17. Run p through each trained model and get the predictions:  18. Compute the ensemble prediction using current weights:     19. If , then raise alert to Hospital Authorities.  **20. End For** |

Following this, standard scaling was applied to all numerical features to normalize the data and bring the features onto a comparable scale. This step is essential for improving the convergence and performance of neural network models such as FLE-CNN. Next, one-hot encoding was used to transform categorical features into a suitable numerical representation. Since many machine learning models, including convolutional neural networks, require numerical input, this transformation enabled categorical attributes such as protocol types and service types to be included in the model training. Also, label encoding was done to the target variable to prepare it for ML tasks.

|  |
| --- |
| **Algorithm 2:** Federated Ensemble Aggregation and Global Update (Cloud Server) |
| **Input:** Weights from all Hospital clients **Output:** Global ensemble weights ​  **1. For each communication round r = 1 to R: 2.**Receive local weights of all models from all hospital clients Hi ∈ H. 2. Take average of each of the model weights sent by all Hospital Clients which are the global weights.  3. Send the calculated global weights of each model to all the Hospital clients.  4. Clients update their local ensemble model with the global weights. 4. **End For** |
|  |

To reduce the dimensionality of the dataset and improve computational efficiency, Principal Component Analysis (PCA) was employed. PCA helped retain the most informative components while discarding less significant features, thus preserving maximum variance in a lower-dimensional space. This step was particularly beneficial in enhancing the generalization ability of the FLE-CNN model. The entire data preparation pipeline was structured and implemented as described in Algorithm 1. This included the division of the dataset into features (X) and labels (y), followed by a 70/30 split for training and testing. The training data was used to fit scalers and PCA transformers, which were then applied to the testing data to ensure consistency. Afterward, the preprocessed training set was used to train the FLE-CNN model, and the performance was evaluated on the testing set. Where necessary, hyperparameter tuning was conducted to optimize the model.

*C. Federated Learning Framework:*

In the proposed Federated Learning Ensemble CNN framework, each hospital functions as an independent client with its own local dataset. Instead of sharing sensitive patient data, each client performs local training using various machine learning models - Support Vector Machine (SVM), K-Nearest Neighbours (KNN), Decision Tree (DT), and Convolutional Neural Network (CNN). The data undergoes preprocessing steps including duplication removal, feature scaling, encoding, and dimensionality reduction using Principal Component Analysis. Once trained, the performance of each model is evaluated locally, and their accuracies are used to compute normalized ensemble weights that reflect their individual contribution to the final prediction.

These local model weights are then securely transmitted to a central cloud server for aggregation, where global ensemble weights are constructed based on the collected client updates. This decentralized approach ensures data privacy while enabling collaborative model improvement across distributed medical institutions. Additionally, this methodology enhances the robustness of models by capturing diverse weights from multiple sources, which may not be present in a single dataset. As a result, the federated ensemble learning framework contributes to a more generalized and accurate intrusion detection system, particularly suitable for sensitive domains like healthcare where data confidentiality is paramount.

# Table II Details Of Simulation Parameters

| **Parameter** | **Value** |
| --- | --- |
| Operating system | Microsoft Windows(R) 11 Home |
| Platform used | Visual Studio Code, Jupyter Notebook |
| Processor | 8 × Intel(R) i5 CPU @ 2.20GHz |
| GPU | 4 GB Intel(R) Xeon Graphics |
| Random access memory (RAM) | 8 GB |
| Programming language | Python 3.8 |
| Used libraries | Scikit-learn, Scapy, Numpy |
| Machine learning models | Support Vector Machine (SVM), K-NearestNeighbours(KNN), Decision Tree, Convolutional Neural Network (CNN) |

# V. Practical Implementation

This section describes the practical execution setup of the proposed federated intrusion detection framework, outlining the hardware specifications and software tools employed. The implementation was carried out on a machine running Microsoft Windows 11 Home, equipped with an Intel(R) i5 processor featuring 8 logical cores, operating at 2.20 GHz, and supported by 8 GB of RAM. The graphical processing was handled by an integrated 4 GB Intel Xeon Graphics unit.

Development and simulation were performed using Visual Studio Code and Jupyter Notebook environments. The programming was done in Python 3.8, utilizing prominent libraries such as Scikit-learn for machine learning, Scapy for packet manipulation, and NumPy for numerical computations. Detailed simulation parameters are presented in Table II.

A range of machine learning algorithms was deployed on the local client datasets, including Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Decision Tree (DT), and Convolutional Neural Network (CNN), to train individual models at each participating client.

*Evaluation Metrics:*

To assess the performance of the proposed system, four primary evaluation metrics were considered: True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN). TP indicates instances where malicious activities were correctly detected as threats, while TN denotes cases where benign traffic was accurately recognized as safe. Conversely, FP accounts for situations where normal behaviour was wrongly flagged as malicious, and FN represents instances where actual threats were incorrectly classified as normal.

*Accuracy***:** Measures the proportion of correctly predicted instances among all observations.

*Recall (Sensitivity):* Indicates the proportion of actual positives that were identified correctly.

*Precision:* Reflects the proportion of positive predictions that were indeed correct.

*F1-Score:* Represents the harmonic mean of precision and recall, giving a balanced measure of accuracy in imbalanced datasets.

These metrics were used to comprehensively evaluate the detection capabilities of the proposed system and ensure its effectiveness in identifying cyber threats with high precision and reliability.

\*\*\*Pending – Comparisions, Final Graphs, Results, Conclusion, Refernces

# V. Conclusion

## A conclusion section is not required. Although a conclusion may review the main points of the article, do not replicate the abstract as the conclusion. A conclusion might elaborate on the importance of the work or suggest applications and extensions.

Appendix

Appendixes, if needed, appear before the acknowledgment.

Acknowledgment

The preferred spelling of the word “acknowledgment” in American English is without an “e” after the “g.” Use the singular heading even if you have many acknowledgments. Avoid expressions such as “One of us (S.B.A.) would like to thank ... .” Instead, write “F. A. Author thanks ... .” In most cases, sponsor and financial support acknowledgments are placed in the unnumbered footnote on the first page, not here.

# References and Footnotes

## A. References

References need not be cited in text. When they are, they appear on the line, in square brackets, inside the punctuation. Multiple references are each numbered with separate brackets. When citing a section in a book, please give the relevant page numbers. In text, refer simply to the reference number. Do not use “Ref.” or “reference” except at the beginning of a sentence: “Reference [3] shows ... .” Please do not use automatic endnotes in *Word*, rather, type the reference list at the end of the paper using the “References” style.

Reference numbers are set flush left and form a column of their own, hanging out beyond the body of the reference. The reference numbers are on the line, enclosed in square brackets. In all references, the given name of the author or editor is abbreviated to the initial only and precedes the last name. Use them all; use *et al*. only if names are not given or if there are more than 6 authors. Use commas around Jr., Sr., and III in names. Abbreviate conference titles. When citing IEEE Transactions, provide the issue number, page range, volume number, month if available, and year. When referencing a patent, provide the day and the month of issue, or application. References may not include all information; please obtain and include relevant information. Do not combine references. There must be only one reference with each number. If there is a URL included with the reference, it can be included at the end of the reference.

Other than books, capitalize only the first word in an article title, except for proper nouns and element symbols. For articles published in translation journals, please give the English citation first, followed by the original foreign-language citation. See the end of this document for formats and examples of common references. For a complete discussion of references and their formats, see the *IEEE Editorial Style Manual* *for Authors* at <https://journals.ieeeauthorcenter.ieee.org/create-your-ieee-journal-article/create-the-text-of-your-article/ieee-editorial-style-manual/>.

## B. Footnotes

Number footnotes separately in superscripts (Insert | Footnote).[[1]](#footnote-1) Place the actual footnote at the bottom of the column in which it is cited; do not put footnotes in the reference list (endnotes). Use letters for table footnotes (see Table I).

# Submitting Your Article for Review

## A. Review Stage Using ScholarOne Manuscripts

Contributions to the Transactions, Journals, and Letters may be submitted electronically on IEEE’s online manuscript submission and peer-review system, ScholarOne Manuscripts. You can get help choosing the correct publication for your manuscript as well as find their corresponding ScholarOne Manuscripts peer review site using the tools listed at <http://www.ieee.org/publications_standards/publications/authors/authors_submission.html> Once you have chosen your publication and navigated to the ScholarOne site, check first to see if you have an existing account. If there is none, please create a new account. After logging in, go to your Author Center and click “Start New Submission.”

Along with other information, you will be asked to select the manuscript type from the journal’s pre-determined list of options. Depending on the journal, there are various steps to the submission process; please make sure to carefully answer all of the submission questions presented to you. At the end of each step you must click “Save and Continue”; just uploading the paper is not sufficient. After the last step, you should see a confirmation that the submission is complete. You should also receive an e-mail confirmation. For inquiries regarding the submission of your paper on ScholarOne Manuscripts, please contact oprs-support@ieee.org or call +1 732 465 5861.

ScholarOne Manuscripts will accept files for review in various formats. There is a “Journal Home” link on the log-in page of each ScholarOne Manuscripts site that will bring you to the journal’s homepage with their detailed requirements; please check these guidelines for your particular journal before you submit.

## B. Final Stage Using ScholarOne Manuscripts

Upon acceptance, you will receive an email with specific instructions regarding the submission of your final files. To avoid any delays in publication, please be sure to follow these instructions. Final submissions should include source files of your accepted manuscript, high quality graphic files (if not embedded in your source file), and a formatted pdf file. The accepted version of your manuscript will also be sent to the IEEE publication teams for a comparison to the final files to ensure no significant or unauthorized changes were made after acceptance. If you have any questions regarding the final submission process, please contact the administrative contact for the journal.

When submitting your final files on a hybrid OA journal you will have the opportunity to designate your article as “open access” if you agree to pay the IEEE open access fee. Please select the appropriate choice. Immediately after you have submitted your final files through ScholarOne Manuscripts you will be automatically redirected to the IEEE electronic copyright form wizard. Please complete the copyright at that time to avoid publication delays.

## C. Copyright Form

Authors must submit an electronic IEEE Copyright Form (eCF) upon submitting their final manuscript files. You can access the eCF system through your manuscript submission system or through the Author Gateway. You are responsible for obtaining any necessary approvals and/or security clearances. For additional information on intellectual property rights, visit the IEEE Intellectual Property Rights department web page at <https://www.ieee.org/publications/rights/index.html>

# IEEE Guidelines and Policies

A full overview of IEEE publishing guidelines and policies can be found at <https://journals.ieeeauthorcenter.ieee.org/become-an-ieee-journal-author/publishing-ethics/guidelines-and-policies/>. They are designed to help authors understand and navigate the publishing process successfully. Learn more about IEEE’s fundamental publishing guidelines and principles, submission and peer review policies, post-publication policies, and guidelines on advertising, accessibility, and data privacy.

References

[1] A. Belenguer, J. Navaridas, and J. A. Pascual, "A review of Federated Learning in Intrusion Detection Systems for IoT," *arXiv preprint arXiv:2204.12443*, 2022.

[2] S. Agrawal et al., "Federated Learning for Intrusion Detection System: Concepts, Challenges and Future Directions," *arXiv preprint arXiv:2106.09527*, 2021.

[3] A. M. Abdel-Basset et al., "FIDChain: Federated Intrusion Detection System for Blockchain-Enabled IoT Healthcare Applications," *Healthcare*, vol. 10, no. 6, p. 1110, 2022.

[4] A. Almaghthawi et al., "Federated-Learning Intrusion Detection System Based Blockchain Technology," *International Journal of Online and Biomedical Engineering (iJOE)*, vol. 20, no. 11, pp. 16–30, 2024.

[5] V. T. Nguyen and R. Beuran, "FedMSE: Federated learning for IoT network intrusion detection," *arXiv preprint arXiv:2410.14121*, 2024.

[6] M. Gourceyraud et al., "Federated Intrusion Detection System Based on Unsupervised Machine Learning," *arXiv preprint arXiv:2503.22065*, 2025.

[7] A. Almarashdeh et al., "Privacy-Preserving Federated Learning-Based Intrusion Detection System for IoHT Devices," *Electronics*, vol. 14, no. 1, p. 67, 2025.

[8] Riccardo Lazzarini et al., “Federated Learning for IoT Intrusion Detection”, AI 2023, 4(3), 509-530; https://doi.org/10.3390/ai4030028

[9] “NSL-KDD dataset.” Accessed: Apr. 2025. [Online]. Available: http://nsl.cs.unb.ca/nsl-kdd/

*Basic format for books:*

J. K. Author, “Title of chapter in the book,” in *Title of Published Book, x*th ed. City of Publisher, (only U.S. State), Country: Abbrev. of Publisher, year, ch. x, sec. *x*, pp. xxx–xxx*.*

*Examples:*

1. G. O. Young, “Synthetic structure of industrial plastics,” in *Plastics,* 2nd ed., vol. 3, J. Peters, Ed. New York, NY, USA: McGraw-Hill, 1964, pp. 15–64.
2. W.-K. Chen, *Linear Networks and Systems.* Belmont, CA, USA: Wadsworth, 1993, pp. 123–135.
3. Philip B. Kurland and Ralph Lerner, eds., *The Founders’ Constitution.* Chicago, IL, USA: Univ. of Chicago Press, 1987, Accessed on: Feb. 28, 2010, [Online]. Available: http://press-pubs.uchicago.edu/founders/

*Basic format for handbooks:*

*Name of Manual/Handbook, x* ed., Abbrev. Name of Co., City of Co., Abbrev. State, Country, year, pp. xxx-xxx.

*Examples:*

1. *Transmission Systems for Communications*, 3rd ed., Western Electric Co., Winston-Salem, NC, USA, 1985, pp. 44–60.
2. *Motorola Semiconductor Data Manual*, Motorola Semiconductor Products Inc., Phoenix, AZ, USA, 1989.
3. R. J. Hijmans and J. van Etten, “Raster: Geographic analysis and modeling with raster data,” R Package Version 2.0-12, Jan. 12, 2012. [Online]. Available: http://CRAN.R-project.org/package=raster

*Basic format for reports:*

J. K. Author, “Title of report,” Abbrev. Name of Co., City of Co., Abbrev. State, Country, Rep. xxx, year.

*Example:*

1. E. E. Reber, R. L. Michell, and C. J. Carter, “Oxygen absorption in the earth’s atmosphere,” Aerospace Corp., Los Angeles, CA, USA, Tech. Rep. TR-0200 (4230-46)-3, Nov. 1988.

*Basic format for conference proceedings:*

J. K. Author, “Title of paper,” in *Abbreviated Name of Conf.*, City of Conf., Abbrev. State (if given), Country, year, pp. xxxxxx*.*

*Examples:*

1. D. B. Payne and J. R. Stern, “Wavelength-switched passively coupled single-mode optical network,” in *Proc. IOOC-ECOC,* Boston, MA, USA,1985,   
   pp. 585–590.
2. D. Ebehard and E. Voges, “Digital single sideband detection for interferometric sensors,” presented at the 2nd Int. Conf. Optical Fiber Sensors*,* Stuttgart, Germany, Jan. 2-5, 1984.
3. PROCESS Corporation, Boston, MA, USA. Intranets: Internet technologies deployed behind the firewall for corporate productivity. Presented at INET96 Annual Meeting. [Online]. Available: http://home.process.com/Intranets/wp2.htp

*Basic format for electronic documents (when available online):*

Issuing Organization. (year, month day). *Title*. [Type of medium]. Available: site/path/file

*Example:*

1. U.S. House. 102nd Congress, 1st Session. (1991, Jan. 11). *H. Con. Res. 1, Sense of the Congress on Approval of Military Action*. [Online]. Available: LEXIS Library: GENFED File: BILLS

*Basic format for patents:*

J. K. Author, “Title of patent,” U.S. Patent *x xxx xxx*, Abbrev. Month, day, year.

*Example:*

1. G. Brandli and M. Dick, “Alternating current fed power supply,” U.S. Patent 4 084 217, Nov. 4, 1978.

*Basic format**for theses (M.S.) and dissertations (Ph.D.):*

J. K. Author, “Title of thesis,” M.S. thesis, Abbrev. Dept., Abbrev. Univ., City of Univ., Abbrev. State, year.

J. K. Author, “Title of dissertation,” Ph.D. dissertation, Abbrev. Dept., Abbrev. Univ., City of Univ., Abbrev. State, year.

*Examples:*

1. J. O. Williams, “Narrow-band analyzer,” Ph.D. dissertation, Dept. Elect. Eng., Harvard Univ., Cambridge, MA, USA, 1993.
2. N. Kawasaki, “Parametric study of thermal and chemical nonequilibrium nozzle flow,” M.S. thesis, Dept. Electron. Eng., Osaka Univ., Osaka, Japan, 1993.

*Basic format for the most common types of unpublished references:*

J. K. Author, private communication, Abbrev. Month, year.

J. K. Author, “Title of paper,” unpublished.

J. K. Author, “Title of paper,” to be published.

*Examples:*

1. A. Harrison, private communication, May 1995.
2. B. Smith, “An approach to graphs of linear forms,” 2014, *arXiv:2105.02824*.
3. A. Brahms, “Representation error for real numbers in binary computer arithmetic,” IEEE Computer Group Repository, Paper R-67-85.

*Basic formats for standards:*

a) *Title of Standard*, Standard number, date.

b) *Title of Standard*, Standard number, Corporate author, location, date.

*Examples:*

1. IEEE Criteria for Class IE Electric Systems, IEEE Standard 308, 1969.
2. Letter Symbols for Quantities, ANSI Standard Y10.5-1968.

**First A. Author** (Fellow, IEEE) and all authors may include biographies if the publication allows. Biographies are often not included in conference-related papers. Please check the Information for Authors to confirm. Author photos should be current, professional images of the head and shoulders. The first paragraph may contain a place and/or date of birth (list place, then date). Next, the author’s educational background is listed. The degrees should be listed with the type of degree in what field, which institution, city, state, and country, and year the degree was earned. The author’s major field of study should be lowercase. 

The second paragraph uses the preferred third person pronoun (he, she, they, etc.) and not the author’s last name. It lists military and work experience, including summer and fellowship jobs. Job titles are capitalized. The current job must have a location; previous positions may be listed without one. Information concerning previous publications may be included. The format for listing publishers of a book within the biography is: *Title of Book* (publisher name, year) similar to a reference. Current and previous research interests end the paragraph.

The third paragraph begins with the author’s preferred title and last name (e.g., Dr. Smith, Prof. Jones, Mr. Kajor, Ms. Hunter, Mx. Riley). List any memberships in professional societies other than the IEEE. Finally, list any awards and work for IEEE committees and publications.

**Second B. Author**, photograph and biography not available at the time of publication.

**Third C. Author, Jr.** (Member, IEEE), photograph and biography not available at the time of publication.

1. It is recommended that footnotes be avoided (except for the unnumbered footnote with the receipt date on the first page). Instead, try to integrate the footnote information into the text. [↑](#footnote-ref-1)