Deep Learning-Based Network Intrusion Detection And Prevention System

Huang Nana*

Henan College of Arts and Culture, School of Cultural Communication, Zhengzhou, Henan Province,451464, China

Corresponding author. E-mail: 13937197635@163.com

Received: Feb. 16, 2026; Accepted: Mar. 27, 2026

Current network intrusion detection systems struggle with feature representation, unknown attack detection, and coordinated response. This paper proposes an intelligent system that fuses NetFlow and payload features, employs a three-level detection engine (deep autoencoder, Transformer, GNN), and integrates with software-defined networking for real-time mitigation and adaptive feedback-driven model improvement. Experiments on a mixed dataset combining the CIC-IDS2018 and UNSW-NB15 show a detection rate of 98.7%, a false positive rate of 0.86%, and an average detection rate of 87.04% for unknown attacks, with real-time interception success reaching 99%.

Keywords: Network Intrusion Detection and Prevention; Feature Fusion; Graph Neural Network; Software-Defined Network

http://dx.doi.org/10.6180/jase.202609_32.013

1. Introduction

This study presents a deep-fusion NIDPS integrating traffic and payload features with a three-level SDN-based detection engine for real-time response, supported by incremental learning for continuous adaptation to evolving threats. such as Thockchom et al. [1] demonstrate improved detection accuracy and stability. Ravi et al. [2] proposed a deep learningbased IDS for Internet of Medical Things that combines network and biometric features with attention mechanisms, achieving up to 99% detection accuracy. Biyyapu et al. [3] Extracted discriminative traffic features with feature aggregation and balanced training data via mixed sampling. IoT growth poses unique security challenges; K-NN [4] and Bi-LSTM with XAI [5] have been proposed for efficient intrusion detection. Induru et al. [6] [7, 8] used LSTM to analyze sequential network traffic for detecting malicious activities, a concept integrated into the proposed deep learning-based intrusion detection system to improve accuracy and identify complex attacks. Apruzzese et al. [9] cross-evaluated various machine learning base IDS. Mohamed and Ejbali [10] used deep SARSA for anomaly detection, [11, 12] while Ravi et al. [13] applied deep learning for IoMT intrusion detection. Meta-classifiers [14] improve base classifier performance, and XAI [15] enhances model interpretability, though a fully closedloop active defense system remains lacking. Most existing approaches treat detection and mitigation as independent processes, reducing their effectiveness against dynamic threats.

2. Materials and methods

2.1. System Overall Architecture

The system uses an end-to-end architecture with offline feature extraction and threat detection and online SDN-based response supported by a shared database for closed-loop learning Fig. 1 [16–18], Offline analysis periodically updates features and models and synchronizes them with the online engine, while the online module performs real-time

inference with asynchronous updates to reduce latency and maintain consistency.

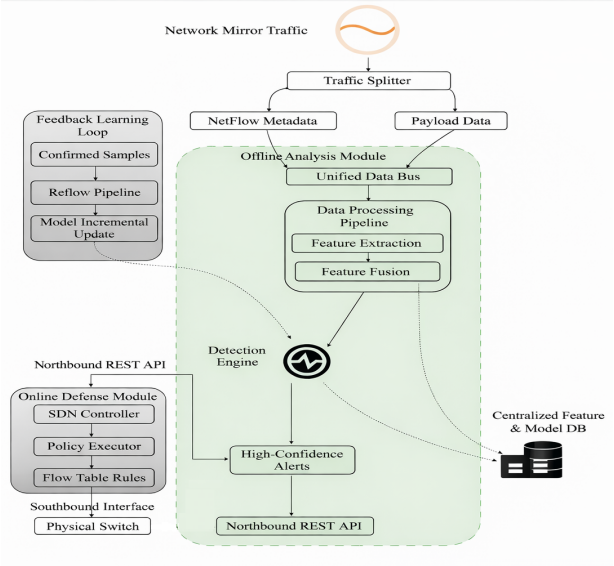

Fig. 1. Network Intrusion Detection and Prevention System

2.2. Data Preprocessing and Fusion Feature Representation Module

The module transforms NetFlow and payloads into standardized numerical vectors for spatiotemporal and semantic analysis. Traffic volume, temporal behavior, and bidirectional flow are prioritized, with correlation and contribution analysis identifying key features validated across attack scenarios. Table 1 lists the standardization methods and numerical transformation ranges of NetFlow feature fields during the preprocessing stage:

A fixed sliding window (10 -second window with 1 -second step) aggregates NetFlow records into session sequences, balancing temporal context and efficiency. Each record is converted into a 15-dimensional feature vector (e.g., packet count, byte count, duration, flag statistics). Session windows form 2D matrices, while payloads are normalized to 1500-byte sequences and processed using a 1D CNN to extract local and semantic features. The convolution operation is defined as follows.

$$C[i,j] = \mathrm{ReLU}\left(\sum_{k=1}^{5} E[i+k-1,:] \cdot W_j[k] + b_j\right) \quad (1)$$

C[i, j] is the output at position iii and channel j; Wj[k] and bj are the convolution weights and bias, with 128 filters and stride 1 . Comparative experiments were conducted using alternative kernel sizes (3,5 , and 7) and filter depths (64,128 , and 256) to evaluate their influence on payload feature extraction performance. NetFlow and payload features are pooled into 128-dimensional vectors and concatenated into a 256-dimensional fused feature vector $f_{fused} \in R^{256}$, which can be described as:

$$f_{fused} = [\mathrm{MaxPool}(X_{flow}); \mathrm{MaxPool}(C)] \quad (2)$$

X_{flow} refers to the NetFlow feature matrix, and C is the convolutional feature map of the payload data. Comparative tests showed that simple concatenation of NetFlow and payload features provides stable detection without added model complexity.

Multi-model Collaborative Detection Engine

The three-level detection engine [19] starts with a DAE having a 256-128-64 encoder and symmetrical decoder, using MSE as the reconstruction loss. Which can be formalized as:

$$L_{recon} = \frac{1}{N}\sum_{i=1}^{N} \|x_i - \hat{x}_i\|^2 \quad (3)$$

A DAE is trained on normal traffic, where inputs with high reconstruction error trigger second-level analysis. The encoder (256-128-64) compresses features while preserving traffic patterns. The second level uses a Transformer with 2 encoder layers, 4 attention heads, and a 512-dim feed-forward network. Ablation tests show this setup balances accuracy and latency. The model output probabilities are computed using the SoftMax function.

$$P(y = c \mid z) = \frac{\exp^2 f\left(w_c^\top z + b_c\right)}{\sum_{j=1}^{J} \exp\left[\exp_j^{fo}\left(w_j^\top z + b_j\right)\right.} \quad (4)$$

The Transformer encoder output is average-pooled and passed through a Softmax layer to predict attack probabilities across J categories, with temperature scaling for calibrated outputs. These scores are fused with DAE and GNN results for consistent detection. The third level uses a GNN on a dynamic host-service graph, where edges represent alarm-related NetFlow sessions. two-layer GCN updates node representations through message passing, and the final graph representation is obtained by weighted averaging of all node features.

$$\alpha_v = \frac{\exp_u \exp_u\left(q^T h_v^{(L)}\right)}{\sum_{u \in V} \exp\left(q^T u^{(L)}\right)} \quad (5)$$

$$h_G = \sum_{v \in V} \alpha_v h_v^{(L)} \quad (6)$$

Table 1. NetFlow feature field standardization parameters

Feature Field	Original Value Range	Standardization Method	Processed Value Range
Packet Count	0-5000	Z-score	-2.5-2.5
Byte Count	0-2000000	Min-Max	0-1
Duration (seconds)	0-3600	Log Scaling	0-8.2
TCP Flag Statistics	0-255	Unitization	0-1
Average Packet Size	0-1500	Z-score	-3.0-3.0

Here $\mathbf{h}_{(}\mathbf{G}\big)$ represents the final graph representation, $\mathbf{V}$ denotes all nodes, and $\mathbf{h_v}^{(\mathbf{L})}$ aggregated via attention $\boldsymbol{\alpha_v}$ to capture abnormal clusters and attack patterns [20].

2.3. Online Prevention and Adaptive Feedback Learning Module

The online module uses an SDN controller to receive high-confidence alarms and block malicious traffic in real time via OpenFlow [21]. Flow rules are prioritized by attack confidence and severity, with conflicts resolved via priority comparison for consistent SDN mitigation. Flow entries have idle timeouts and lifetimes dynamically adjusted based on network load.

$$\mathrm{T_{idle}} = \beta \cdot \frac{1}{\log fo\,(1+\mathrm{R_{alert}})} \tag{7}$$

$\mathrm{T_{idle}}$ represents the final idle timeout setting, β is the baseline time constant, and $\mathrm{R_{alert}}$ is the generation rate of the same type of alarm per unit time.

Table 2 shows how the system dynamically calculates and sets the timeout based on R_{alert} under different network load scenarios. For example, substituting $T_{\mathrm{base}} = 300$ and $R_{\mathrm{alert}} = 0.1$ into Eq. (7) gives 301 seconds, while $T_{\mathrm{base}} = 100$ and $R_{\mathrm{alert}} = 10$ produce 130 seconds, matching the values in Table 2. The value of the baseline time constant β is also finetuned according to the network's basic security policy:

Alarms, predictions, and labels feed an incremental dataset for adaptive learning. Multistage verification removes corrupted records, and credibility scoring filters low-confidence or adversarial samples for reliable incremental learning. DAE, Transformer, and GNN are finetuned on incremental data (lr 0.0001, 5 epochs) to adapt to new threats [22–25].

$$\mathrm{L_{inc}} = \mathrm{L_{new}} + \lambda \cdot \mathrm{L_{memory}} \tag{8}$$

L_{new} is the loss on the newly added data, $\mathrm{L_{memory}}$ is the loss on the memory set sampled from the old data, and λ is a hyperparameter that weighs the importance of both.

Fine-tuning uses a dynamically decaying learning rate updated after each epoch.

According to the following rule:

$$\eta_t = \eta_0 \cdot \gamma^t \tag{9}$$

η_t is the learning rate at the t-th epoch, η_0 is the initial learning rate (0.0001), and γ is the decay coefficient (0.8).

3. Results and discussion

3.1. Experimental Environment and Datasets

The experiment used a hybrid dataset of CIC-IDS2018 and UNSW-NB15 two publicly available and widely recognized benchmark intrusion detection datasets to cover diverse attack scenarios. CIC-IDS2018 and UNSW-NB15 (2,850,000 records, 85 features, 1500-byte payloads) were split 7:2:1; training used known attacks, testing included unknown variants (Table 3). hybrid datasets

3.2. Overall System Performance

DR and FPR were measured on the test set; the proposed system used a three-level detection engine, while baseline models used original outputs. Results are shown in Fig. 2.

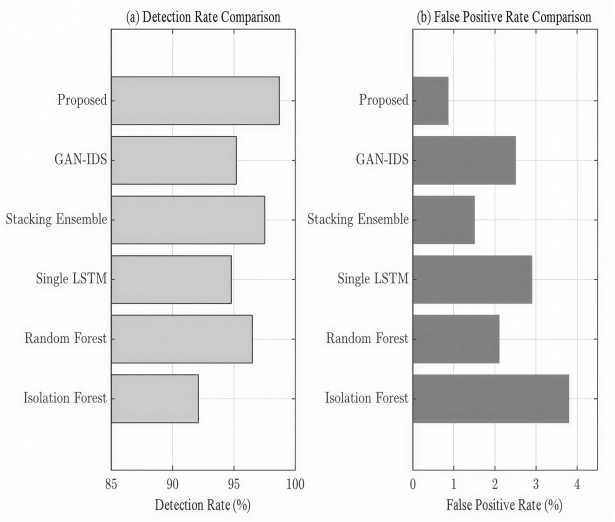

Fig. 2. Detection accuracy and false positive rate comparison of DL-NIDPS and baseline models on the CIC-IDS2018 and UNSW-NB15

Table 2. Dynamic configuration of SDN flow table entry idle timeout

Network Load Scenario	Base Time Constant (seconds)	Alert Rate (per second)	Calculated Idle Timeout (seconds)
Low Load, Regular Policy	300	0.1	301
Low Load, Strict Policy	200	0.5	232
Medium Load, Balanced Policy	180	2.0	152
High Load, Lenient Policy	150	5.0	136
Attack Surge, Emergency Policy	100	10.0	130

Table 3. Details of the composition and partitioning of the CIC-IDS2018 and UNSW-NB15

Data Subset	Number of Attack Types	Number of Records	Normal Traffic (%)	Attack Traffic (%)
Training Set	23 (Known attacks)	1,995,000	75.2	24.8
Validation Set	23 (Known attacks)	570,000	70.5	29.5
Test Set	23 (20 Known + 3 Unknown variants)	285,000	65.0	35.0
Total	26	2,850,000	72.8	27.2

Experiments on the combined CIC-IDS2018 and UNSW-NB15 dataset show that DLNIDPS achieves a 98.7% detection rate and 0.86% false positive rate. Multiple runs with different seeds confirm reproducible performance, improving detection by 1.2% and halving FPR compared to Stacking, while maintaining real-time latency (Fig. 3).

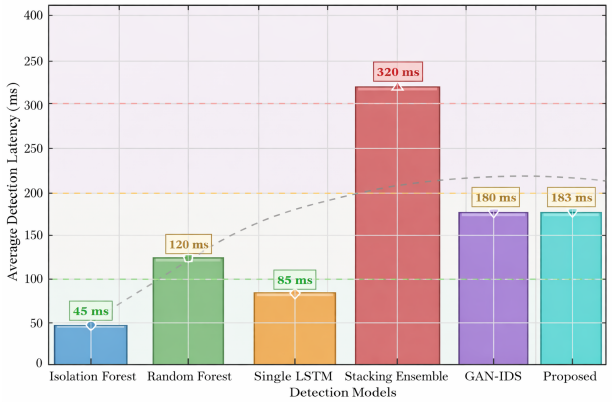

Fig. 3. Average inference latency of the proposed DL-NIDPS compared with baseline models under the same experimental environment

The latency values represent the average per-sample inference time measured over 10,000 consecutive inputs under identical hardware conditions. Average latency is 183 ms ; the cascaded design balances speed and accuracy. Table 4 shows DR, FPR, and F1 for full and ablated modules.

The system achieved 98.9% F1-score. Ablation experiments (five runs with different seeds) show stable results with low deviation, confirming reliability. Results indicate that DAE, Transformer, and GNN are all critical, with the three-level design effectively balancing high detection rate (DR) and low false positive rate (FPR).

3.3. Zero-Day Attack Detection Capability Evaluation

Five test rounds with different unknown attacks were conducted, and each model's detection rates are shown in Fig. 4. Per-class metrics, confusion matrices, and variance analysis assessed zero-day adaptability and stability, while uncertainty and distribution-shift tests confirmed robust performance.

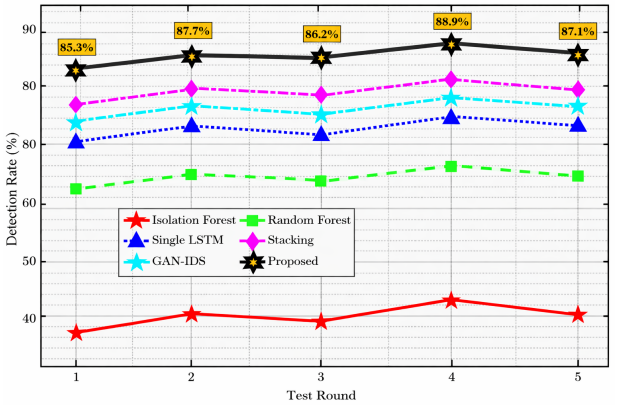

Fig. 4. Detection performance of different models across five zero-day attack test rounds showing the stability of the proposed system

The proposed system maintained $\sim$ 87% stable detection across five rounds, outperforming other models, while Isolation Forest was unstable.

3.4. Online Defense Effectiveness

1000 diverse attack sessions were injected to evaluate interception performance, and the interception rate was calculated by comparing injected attacks with packets received. by the server (Fig. 5).

The system blocked 99% of brute-force/port-scan and 96.6-97% of DDoS/web attacks, showing effective adaptive defense. Accurate detection with low false positives

Table 4. DR, FPR, and F1 for full and ablated modules

Ablation Configuration	Detection Rate (DR, %)	False Positive Rate (FPR, %)	F1-Score (%)
Full DL-NIDPS	98.7	0.86	98.9
Remove DAE (Stage 1)	97.9	2.4	97.6
Remove Transformer (Stage 2)	90.2	1.1	93.8
Remove GNN (Stage 3)	98.4	1.5	98.5

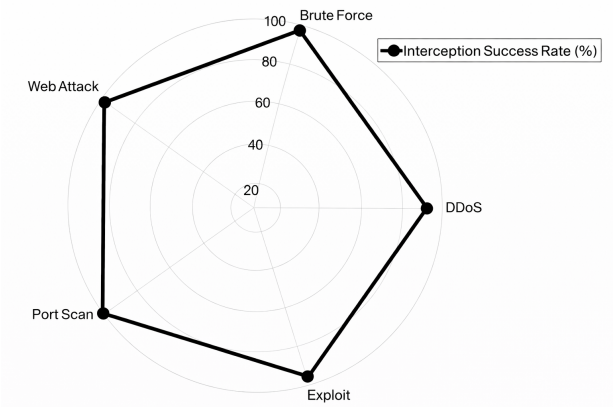

Fig. 5. Interception success rate of the SDN-based defense module for different attack types during simulated attack sessions

enables timely mitigation, maintaining service availability and reducing financial losses.

4. Conclusions

This paper presents a three-level detection engine using DAE, Transformer, and GNN on NetFlow and payload features and establishes an intelligent intrusion detection and prevention system with SDN-enabled linked response mechanisms. The system solves feature, imbalance, and fragmentation issues, while future work should focus on encrypted traffic, adversarial robustness, and real-world deployment.

5. Acknowledgment

Not applicable.

6. Declaration

7. Funding

There is no specific funding to support this research.

8. Conflict of interest

The authors declare that they have no conflicts of interest regarding this work.

9. Data availability

The data that support the findings of this study are not publicly available due to confidentiality agreements but are available from the corresponding author upon reasonable request.

10. Code availability

Not applicable.

11. Author contributions

Huang Nana contributed to the design and methodology of this study, the assessment of the outcomes, and the writing of the manuscript.

References

[1] N. Thockchom, M. M. Singh, and U. Nandi, (2023) *"A novel ensemble learning-based model for network intrusion detection"* **Complex Intelligent Systems** *9*(5): 5693–5714. DOI: 10.1007/s40747-023-01013-7.

[2] V. Ravi, T. D. Pham, and M. Alazab, (2023) *"Deep learning-based network intrusion detection system for internet of medical things"* **IEEE Internet of Things Magazine** *6*(2): 50–54. DOI: 10.1109/IOTM.001.2300021.

[3] N. S. Biyyapu, E. J. Veerapaneni, P. P. Surapaneni, S. S. Vellela, and R. Vatambeti, (2024) *"Designing a modified feature aggregation model with hybrid sampling techniques for network intrusion detection"* **Cluster Computing** *27*(5): 5913–5931. DOI: 10.1007/s10586-024-04270-4.

[4] M. Mohy-Eddine, A. Guezzaz, S. Benkirane, and M. Azrour, (2023) *"An efficient network intrusion detection model for IoT security using K-NN classifier and feature selection"* **Multimedia Tools and Applications** *82*(15): 23615–23633. DOI: 10.1007/s11042-023-14795-2.

[5] S. Sivamohan and S. S. Sridhar, (2023) *"An optimized model for network intrusion detection systems in Industry 4.0 using XAI-based Bi-LSTM framework"* **Neural Computing and Applications** *35*(15): 11459–11475. DOI: 10.1007/s00521-023-08319-0.

[6] V. Induru and P. N, (2019) *"Enhanced network intrusion detection using long short-term memory for improved security analysis"* **International Journal of Engineering Technology Research Management** ***3***(3): DOI: 10.5281/zenodo.15600702.

[7] P. B. Udas, M. E. Karim, and K. S. Roy, (2022) *"SPIDER: A shallow PCA-based network intrusion detection system with enhanced recurrent neural networks"* **Journal of King Saud University – Computer and Information Sciences** ***34***(10): 10246–10272. DOI: 10.1016/j.jksuci.2022.10.019.

[8] Z. K. Maseer, Q. K. Kadhim, B. Al-Bander, et al., (2024) *"Meta-analysis and systematic review for anomaly network intrusion detection systems: detection methods, datasets, validation methodology, and challenges"* **IET Networks** ***13***(5–6): 339–376. DOI: 10.1049/ntw2.12128.

[9] W. Zhao and Z. Zhao, (2024) *"Providing a hybrid approach to increase the accuracy of intrusion detection systems in computer networks"* **Journal of Engineering and Applied Science** ***71***(1): 123. DOI: 10.1186/s44147-024-00404-y.

[10] G. Apruzzese, L. Pajola, and M. Conti, (2022) *"The cross-evaluation of machine learning-based network intrusion detection systems"* **IEEE Transactions on Network and Service Management** ***19***(4): 5152–5169. DOI: 10.1109/TNSM.2022.3204615.

[11] M. Rashid, J. Kamruzzaman, T. Imam, et al., (2022) *"A tree-based stacking ensemble technique with feature selection for network intrusion detection"* **Applied Intelligence** ***52***(9): 9768–9781. DOI: 10.1007/s10489-021-02968-1.

[12] M. Maddu and Y. N. Rao, (2024) *"Network intrusion detection and mitigation in SDN using deep learning models"* **International Journal of Information Security** ***23***(2): 849–862. DOI: 10.1007/s10207-023-00771-2.

[13] S. Mohamed and R. Ejbali, (2023) *"Deep SARSA-based reinforcement learning approach for anomaly network intrusion detection system"* **International Journal of Information Security** ***22***(1): 235–247. DOI: 10.1007/s10207-022-00641-0.

[14] V. Ravi, T. D. Pham, and M. Alazab, (2023) *"Deep learning-based network intrusion detection system for Internet of Medical Things"* **IEEE Internet of Things Magazine** ***6***(2): 50–54. DOI: 10.1109/IOTM.001.2200183.

[15] M. Ali, M. Haque, M. H. Durad, A. Usman, S. M. Mohsin, H. Mujlid, et al., (2023) *"Effective network intrusion detection using stacking-based ensemble approach"* **International Journal of Information Security** ***22***(6): 1781–1798. DOI: 10.1007/s10207-023-00718-7.

[16] P. Barnard, N. Marchetti, and L. A. DaSilva, (2022) *"Robust network intrusion detection through explainable artificial intelligence (XAI)"* **IEEE Networking Letters** ***4***(3): 167–171. DOI: 10.1109/LNET.2022.3186589.

[17] M. Mehmood, T. Javed, J. Nebhen, et al., (2022) *"A hybrid approach for network intrusion detection"* **CMC–Computers, Materials Continua** ***70***(1): 91–107. DOI: 10.32604/cmc.2024.054966.

[18] S. Das, S. Saha, A. T. Priyoti, et al., (2021) *"Network intrusion detection and comparative analysis using ensemble machine learning and feature selection"* **IEEE Transactions on Network and Service Management** ***19***(4): 4821–4833. DOI: 10.1109/TNSM.2021.3138457.

[19] R. Ghanbarzadeh, A. Hosseinalipour, and A. Ghaffari, (2023) *"A novel network intrusion detection method based on metaheuristic optimisation algorithms"* **Journal of Ambient Intelligence and Humanized Computing** ***14***(6): 7575–7592. DOI: 10.1007/s12652-023-04571-3.

[20] Y. S. Almutairi, B. Alhazmi, and A. A. Munshi, (2022) *"Network intrusion detection using machine learning techniques"* **Advances in Science and Technology Research Journal** ***16***(3): 193–206. DOI: 10.12913/22998624/149934.

[21] S. S. Md, (2024) *"A comparative analysis of network intrusion detection using artificial intelligence techniques for increased network security"* **International Journal** ***13***(2): 4014–4025. DOI: 10.30574/ijsra.2024.13.2.2664.

[22] I. Ortega-Fernandez, M. Sestelo, J. C. Burguillo, et al., (2024) *"Network intrusion detection system for DDoS attacks in ICS using deep autoencoders"* **Wireless Networks** ***30***(6): 5059–5075. DOI: 10.1007/s11276-022-03214-3.

[23] C. Park, J. Lee, Y. Kim, et al., (2022) *"An enhanced AI-based network intrusion detection system using generative adversarial networks"* **IEEE Internet of Things Journal** ***10***(3): 2330–2345. DOI: 10.1109/JIOT.2022.3211346.

[24] M. Amru, R. J. Kannan, E. N. Ganesh, et al., (2024) *"Network intrusion detection system by applying ensemble model for smart home"* **International Journal of Electrical and Computer Engineering** ***14***(3): 3485–3494. DOI: 10.11591/ijece.v14i3.pp3485-3494.

[25] K. He, D. D. Kim, and M. R. Asghar, (2023) *"Adversarial machine learning for network intrusion detection systems: A comprehensive survey"* **IEEE Communications Surveys Tutorials 25**(1): 538–566. DOI: 10.1109/COMST.2022.3233793.