A Hybrid Security Framework for Web Applications Using Blockchain and Adaptive Adversarial Learning

Han Wu[1,3,*] and Shugong Zhou[2,3]

[1]*Admission and Employment Office, TangShan Normal University, TangShan 063000, China*
[2]*School of Mathematics and Computational Sciences, TangShan Normal University, TangShan 063000, China*
[3]*Tangshan Key Laboratory of Numerical Computation and Simulation, TangShan 063000, China*
E-mail: wuhan202416@163.com; 126052@tstc.edu.cn
[*]*Corresponding Author*

Received 24 February 2025; Accepted 04 April 2025

Abstract

Web applications are increasingly vulnerable to sophisticated cyberattacks, and traditional security methods often fail to address the dynamic nature of modern threats. To tackle these challenges, we propose a novel security model that integrates blockchain technology, deep learning, and adaptive adversarial learning (ARL). This model aims to enhance web application security by ensuring data integrity, enabling intelligent attack detection, and optimizing defense strategies in real time. By combining these advanced technologies, our model offers a scalable and adaptive solution capable of defending against both known and unknown attacks. Experimental results demonstrate that our approach outperforms existing methods, providing superior protection and resilience against a wide range of cyber threats. Our model not only

Journal of Web Engineering, Vol. 24_3, 355–382.
doi: 10.13052/jwe1540-9589.2432
© 2025 River Publishers

improves detection accuracy but also significantly enhances response times and overall defense efficiency. These results highlight the effectiveness of the proposed model in providing robust and efficient protection for web applications, offering significant improvements over traditional methods in handling dynamic and evolving cyber threats.

Keywords: Web application security, blockchain technology, adaptive adversarial learning, attack detection, defense optimization, intrusion detection system.

1 Introduction

With the rapid development of the internet, web applications have become a core component of many online services, ranging from e-commerce to social networking and online payment systems. These applications have revolutionized the way individuals and businesses interact, conduct transactions, and share information. However, as web applications have proliferated, the scale and complexity of cyberattacks have also increased. Distributed denial of service (DDoS) attacks, SQL injections, cross-site scripting (XSS), and other types of attacks have become increasingly sophisticated, threatening the stability of web applications and potentially leading to data breaches, privacy violations, financial losses, and significant damage to organizational reputation [1–3]. The rapid adoption of cloud services, the widespread use of mobile devices, and the growth of the Internet of Things (IoT) have all expanded the attack surface, increasing the vulnerability of web applications to various cyber threats [4–6]. Consequently, enhancing the protection mechanisms of web-based systems, particularly in the areas of threat identification and mitigation, has emerged as a critical concern within the field of information security.

Many web application security solutions rely on rule-based intrusion detection systems (IDSs) to detect known attacks through predefined signatures or rules [7]. These systems are effective against simple, known attacks, but often fail when confronted with novel or evolving attack patterns. Attackers constantly refine their tactics, exploiting zero-day vulnerabilities and obfuscating malicious activities to bypass rule-based detection. As the tactics of cyber attackers become increasingly sophisticated, traditional rule-based systems are insufficient to address complex threats [8, 9]. Consequently, machine learning and deep learning methods for attack detection have progressively replaced rule-based systems, offering enhanced adaptability and

scalability to address evolving attack patterns. Machine learning techniques like decision trees, SVM, and clustering have demonstrated effectiveness in identifying unknown attacks, while deep learning approaches have proven even more powerful, particularly in managing large-scale datasets [10–13]. Deep learning technologies are particularly effective in handling high-dimensional data, where traditional machine learning methods often struggle. These models can autonomously extract features, reducing the need for manual intervention, which is a common challenge in traditional systems. Deep learning has been widely adopted for web attack detection, achieving notable success. These models automate feature extraction and learning, removing the need for manual feature engineering, and efficiently identifying different types of attacks. By automating feature learning, deep learning systems can adapt to new attack vectors, making them highly suitable for web security, especially in environments where new attack types emerge frequently [14–16]. Despite their advantages, most existing deep learning approaches focus primarily on attack detection and lack real-time defense mechanisms. When dealing with sophisticated or novel attacks, these systems tend to be slow in responding and often fail to adjust defense strategies dynamically based on ongoing attack patterns [17–19]. Attackers frequently use evasive techniques like polymorphic attacks, encryption, and obfuscation to evade detection, rendering static models inadequate for handling such adaptive threats [20]. Although detection accuracy has improved, the real challenge lies in ensuring that once an attack is detected, an appropriate defense mechanism is triggered promptly to mitigate its impact. Existing systems often offer static responses, unable to adjust to the evolving nature of attacks in real-time [21]. Furthermore, these systems frequently struggle with distinguishing between legitimate and malicious traffic, which can overwhelm security teams and hinder operational efficiency.

Although deep learning has made significant strides in attack detection, many existing systems are still limited by their inability to dynamically respond to evolving threats. The defensive strategies employed by these systems lack flexibility, making them ill-suited for dynamic security requirements. To address these challenges, there is increasing interest in integrating machine learning models with adaptive defense mechanisms capable of responding in real time, enabling not only the detection of attacks but also the autonomous adjustment of security policies to counteract ongoing threats.

To address these shortcomings, we propose a novel adaptive defense system that combines deep reinforcement learning (DRL) and ARL to enable both attack detection and dynamic defense optimization. Our model

integrates blockchain technology to ensure data integrity and transparency, providing a secure and reliable foundation for real-time attack detection and defense. Specifically, the ARL component continuously generates adversarial samples that challenge the defense system, ensuring the system learns to detect and defend against previously unknown or evolving attacks. Meanwhile, the DRL component autonomously adjusts defense strategies in response to detected attacks, optimizing the system's defenses in real-time.

The contributions of this paper are threefold:

- We design and implement a novel adaptive defense framework that combines deep learning and reinforcement learning to efficiently detect attacks and optimize defense strategies across multiple attack types.
- We propose a dynamic defense mechanism based on reinforcement learning, which adapts defense strategies in real-time based on attack traffic.
- We validate the effectiveness of the proposed method through extensive experiments, demonstrating that our system outperforms existing methods in terms of detection accuracy, response time, and defense efficiency.

2 Related Work

2.1 Traditional Methods for Web Application Security

Web application security is essential for protecting against cyberattacks, unauthorized access, data leaks, and other malicious activities. Traditional web application security methods primarily rely on techniques such as rule matching, access control, and data encryption. IDSs based on predefined rules are one of the earliest technologies used in web security. These systems detect malicious behavior by matching predefined rules or attack signatures, making them highly effective at defending against known attacks. However, their limitation lies in their inability to effectively defend against new or variant attacks, as they are primarily designed to recognize known attack patterns.

Web application firewalls (WAF) are dedicated tools built to safeguard web applications. They analyze HTTP requests and responses to intercept and block common attacks such as SQL injection, XSS, and cross-site request forgery (CSRF) [22, 23]. The main advantage of WAFs is their ease of deployment, allowing for quick defense against known attacks. However, like IDSs, WAFs are also limited in their ability to defend against unknown

attacks, particularly when attack strategies evolve, as traditional rule-based systems cannot quickly adapt to new threats. Access control is a core technique that ensures only authorized users can interact with specific resources in a web application [24]. Traditional role-based access control (RBAC) methods define roles for users and restrict their access rights accordingly [25]. While RBAC is effective in limiting unauthorized access, it struggles to adapt to complex and dynamic security requirements. Static role and permission configurations can become vulnerabilities, particularly in systems with complex user behavior or multiple hierarchical access levels [26]. Data encryption is another critical method for protecting sensitive information within web applications. By applying encryption algorithms, even if data is intercepted during transmission or storage, unauthorized individuals cannot interpret its contents. Common transmission encryption protocols, such as SSL/TLS, ensure secure communication between web applications and users [27]. While encryption provides strong data privacy protection, it introduces performance overhead, and improper key management may pose a potential security risk [28]. Additionally, log management and security auditing are foundational components of web application security. By recording access logs, operation logs, and other critical activities, security personnel can promptly detect abnormal behaviors and trace security events when they occur [29]. However, traditional log analysis methods rely heavily on static rules, which lack the dynamic ability to identify complex attacks and are prone to false positives and missed detections.

Overall, while traditional web application security methods are highly effective against known attacks, they have limitations in defending against new and evolving threats. These methods often lack flexibility and dynamic response mechanisms. As cyberattacks continue to evolve, machine learning and deep learning-based security approaches are becoming an increasingly important focus in the field of web application security.

2.2 Hybrid Models for Attack Detection and Defense

As the methods of network attacks continue to evolve, single defense mechanisms are no longer sufficient to meet the complex and dynamic security demands. Therefore, hybrid models for attack detection and defense have gradually become a key area of research. These frameworks integrate diverse threat identification and mitigation strategies, capitalizing on the unique advantages of each approach to bolster the protection mechanisms of web-based systems.

A frequently employed integrated strategy involves merging data-driven anomaly identification systems, powered by machine learning algorithms, with conventional signature-based protection mechanisms. Machine learning methods can automatically learn attack patterns from traffic data, identifying unknown attack types, while rule-based defense systems can quickly detect and respond to known attacks [30–32]. By combining these, hybrid models can ensure effective defense against known attacks while also improving the recognition of new attacks, especially when dealing with evolving attack variants, providing a more adaptive approach. Additionally, the combination of deep learning techniques with traditional defense mechanisms is another effective hybrid model. Deep learning methods are capable of autonomously learning complex patterns from network traffic, identifying intricate attack behaviors that traditional systems might miss [33]. When integrated with conventional security measures like WAF or IDS, which can perform real-time filtering and monitoring of traffic, this hybrid approach can enhance attack detection accuracy while reducing the load on the defense system [34]. Another hybrid model combines LSTM with rule-based defense technologies [35]. LSTM can capture sequential information in network traffic, identifying potential attack behaviors, especially in cases where traffic patterns are highly dynamic and complex. By integrating a rule-based defense system, these models can respond more quickly while maintaining high efficiency, making them more adaptable to dynamic traffic patterns [36]. Recently, hybrid models that integrate multiple deep learning techniques with traditional methods have gained attention. For example, combining CNN, RNN, and reinforcement learning has shown significant promise in attack detection [37–39]. These models perform feature extraction using deep learning and use reinforcement learning to dynamically adjust defense strategies, thus enabling more flexible and efficient attack defense [40–42]. Despite the strong performance of hybrid models in web application security, they still face certain challenges. For instance, deep learning models, which tend to have high computational costs, may lead to longer response times, particularly in high-volume traffic environments [43, 44]. Optimizing model computational efficiency and real-time response capability remains a challenge for hybrid models. Furthermore, the design and implementation of hybrid models can be complex, requiring a careful balance between different technologies to ensure the system provides effective protection without incurring excessive resource consumption [45].

Hybrid models for attack detection and defense enhance the security of web applications by combining traditional defense techniques with modern

machine learning or deep learning technologies. These models offer greater adaptability and accuracy, particularly in complex and evolving attack scenarios, by continuously learning and adapting to new attack patterns. As technology continues to evolve, hybrid models are expected to be further optimized and widely applied in the face of dynamic attack environments.

3 Method

3.1 Overview of the overall method

To address the increasingly complex security threats faced by web applications, the proposed security architecture combines blockchain technology, deep learning, and ARL to build an efficient, intelligent, and dynamically optimized security defense system through a modular design. The system consists of four core modules: the blockchain backend architecture module, the security mechanism module, and the attack detection and adaptive defense optimization module. The blockchain backend architecture ensures data integrity and transparency through decentralized storage and smart contract management; in high-frequency request environments, alternative solutions such as off-chain storage and sidechains can be considered to optimize performance and reduce latency. The security mechanism module provides protection through encryption algorithms and multi-factor authentication for data transmission and identity verification; additionally, we address how load balancing and caching mechanisms are employed to handle high concurrency without compromising system security. The attack detection and adaptive defense optimization module leverages ARL technology for intelligent attack recognition and real-time defense strategy adjustment. These modules work in close coordination to form a comprehensive security protection system, enhancing the system's ability to defend against both known and unknown attacks. Figure 1 illustrates the overall architecture and the relationships between these modules.

3.2 Blockchain Backend Architecture Module

The blockchain backend architecture module ensures a decentralized, secure environment for web applications, maintaining data integrity, transparency, and immutability. This module leverages blockchain technology to secure data storage and transactions, while also enabling smart contract execution for identity management and access control. The underlying structure of this module is based on a hybrid blockchain model, which integrates both public

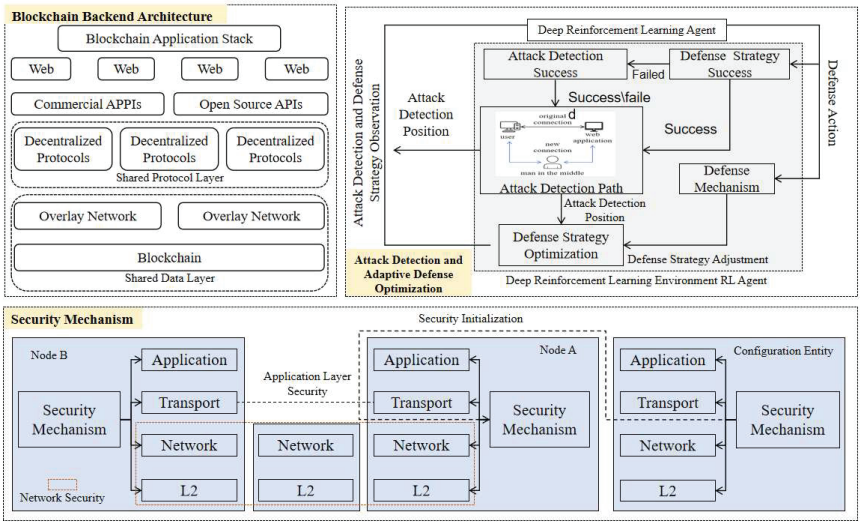

Figure 1 Architecture of the proposed security model.

and private blockchains to offer scalability and confidentiality, along with the decentralized and immutable features of public blockchains. To address high-frequency request environments, the system can incorporate off-chain solutions, such as utilizing sidechains or employing techniques like state channels, to minimize latency and ensure efficient transaction processing.

The architecture of the blockchain system is designed to support high-throughput transactions with minimal latency. It employs a proof of stake (PoS) consensus mechanism, which offers greater energy efficiency and scalability than traditional proof of work (PoW). The PoS algorithm allocates validation power based on the amount of cryptocurrency staked, ensuring that validators have a vested interest in preserving the system's integrity.

Let σ_{i} represent the stake held by participant i, the probability of participant i being selected as the next block validator is given by the formula:

$$\mathrm{P(i)} = \frac{\sigma_{\mathrm{i}}}{\sum_{\mathrm{i}=1}^{\mathrm{n}} \sigma_{\mathrm{i}}} \tag{1}$$

where n is the total number of participants, and $\sum_{\mathrm{i}} = 1^{\mathrm{n}}\sigma_{\mathrm{i}}$ is the total stake in the network. This formula ensures that validators with higher stakes are more likely to be selected, providing an incentive for honest behavior and increasing the security of the blockchain.

In addition to PoS, the blockchain system implements smart contracts to manage decentralized identity (DID) and access control. These contracts are deployed on the blockchain and executed automatically when predefined conditions are met, eliminating the need for trusted intermediaries. Let $\mathcal{C}$ denote a smart contract with conditions $\mathrm{C}_1, \mathrm{C}_2, \ldots, \mathrm{C}_\mathrm{k}$, where each C_i is a condition that must be satisfied for the contract to be executed. The smart contract can be described as:

$$\mathcal{C} = \{\mathrm{C}_1, \mathrm{C}_2, \ldots, \mathrm{C}_\mathrm{k}\} \tag{2}$$

The blockchain backend also supports decentralized data storage to enhance the security and availability of application data. Using a system like the IPFS (InterPlanetary File System), files are stored across multiple nodes, and the integrity of data is ensured through cryptographic hashing. Each file stored on the IPFS is assigned a unique identifier, or content identifier (CID), which is derived from the cryptographic hash of the file's contents:

$$\mathrm{CID} = \mathrm{H}(\text{file contents}) \tag{3}$$

where $\mathrm{H}(\cdot)$ represents a cryptographic hash function. This ensures that any modification to the file would result in a completely different CID making it easy to verify the integrity of the stored data.

Moreover, to facilitate scalability and faster transaction throughput, the blockchain system integrates sidechains. Sidechains are distinct blockchains linked to the main chain (or parent blockchain) via a two-way peg, enabling secure asset transfers between the main chain and sidechains. This provides flexibility by enabling the main chain to handle only high-priority transactions while offloading less critical transactions to the sidechains. The two-way peg mechanism ensures that assets transferred between the main chain and sidechain remain secure and verifiable. Let A represent an asset on the main chain and A' represent the corresponding asset on the sidechain. The transfen from the main chain to the sidechain can be represented by the formula

$$\mathrm{A} \xrightarrow{\text{transfer}} \mathrm{A}' \tag{4}$$

where the transfer occurs when both the main chain and sidechain nodes agree on the validity of the transfer, ensuring the consistency of asset ownership.

The security of the blockchain is further enhanced by the use of cryptographic algorithms such as RSA and elliptic curve cryptography (ECC) for securing transactions and verifying identities. These cryptographic methods

provide robust mechanisms for ensuring the confidentiality and integrity of the data exchanged between participants on the blockchain. For a public-key encryption system like RSA, the encryption of a message M using public key $\mathrm{Pub_k}$ is given by

$$\mathrm{E}(\mathrm{Pub_k}, \mathrm{M}) = \mathrm{M^e} \bmod \mathrm{n} \tag{5}$$

where e and n are components of the public key. The corresponding decryption with private key $\mathrm{Piv_k}$ is given by

$$\mathrm{D}(\mathrm{Priv_k}, \mathrm{E}) = \mathrm{E^d} \bmod \mathrm{n} = \mathrm{M} \tag{6}$$

where d is the private key and M is the original message.

By utilizing these blockchain technologies, this module ensures that the web application's data remains secure, transparent, and immutable, while simultaneously enabling decentralized identity management and access control. The combination of PoS consensus, smart contracts, cryptographic data storage, and sidechain scalability makes the blockchain backend highly efficient, secure, and adaptable to the growing needs of modern web applications.

3.3 Security Mechanism Module

By utilizing these blockchain technologies, this module ensures that the web application's data remains secure, transparent, and immutable, while simultaneously enabling decentralized identity management and access control. The combination of PoS consensus, smart contracts, cryptographic data storage, and sidechain scalability makes the blockchain backend highly efficient, secure, and adaptable to the growing needs of modern web applications.

The security mechanism module provides a multi-layered defense system for web applications, covering the network layer, application layer, and data layer to ensure that the system can effectively counter both internal and external security threats. This module integrates modern encryption technologies, identity verification mechanisms, zero-trust architecture, and dynamic defense strategies to offer comprehensive protection for the web application.

At the network layer, the system deploys a WAF and DDoS protection to prevent common network attacks. The WAF analyzes HTTP requests and matches them against known attack patterns to block malicious traffic. To handle high-frequency attacks, traffic filtering can be optimized using rate-limiting algorithms and edge caching to reduce server load and prevent malicious requests from overwhelming the system. Let T represent the traffic data of a request, and M be the attack pattern database, then the defense

decision can be expressed as:

$$\mathrm{f}(\mathrm{T},\mathrm{M}) = \begin{cases} 1, & \text{if } \mathrm{T} \in \mathrm{M} \\ 0, & \text{if } \mathrm{T} \notin \mathrm{M} \end{cases} \tag{7}$$

When $\mathrm{f}(\mathrm{T},\mathrm{M}) = 1$, this indicates that the traffic T is identified as malicious and the request will be blocked.

At the application layer, the system uses multi-factor authentication (MFA) and decentralized identity management (DID) to ensure the security of user identities. MFA combines multiple verification methods, such as passwords, one-time passcodes, and biometric features. The verification process can be represented as:

$$\mathrm{V} = \mathrm{f}(\mathrm{p},\mathrm{c},\mathrm{b}) \tag{8}$$

where p is password verification, c is the one-time passcode, and b is biometric verification. The function f combines these factors, and the user is authenticated only when all conditions are met, i.e., $\mathrm{V} = 1$. For data encryption at the data layer, we use AES-256 to encrypt sensitive data. The data D after AES encryption is given by

$$\mathrm{E}_{\mathrm{AES}-256}(\mathrm{D}) = \mathrm{C} \tag{9}$$

where $\mathrm{E}_{\mathrm{AES}-256}$ represents the encryption operation, and C is the resulting ciphertext. This ensures that the data is secure both during storage and transmission. In zero trust architecture (ZTA), no user or device is trusted by default. Every access request to the system must undergo authentication and authorization. For each incoming request R, the validation is represented by:

$$\mathrm{V}_{\mathrm{ZT}}(\mathrm{R}) = \mathrm{f}(\mathrm{I},\mathrm{A}) \tag{10}$$

where I represents identity validation and A represents access control policies. Only when both identity validation and access control are passed, $\mathrm{V}_{\mathrm{ZT}} = 1$, will access be granted.

Regarding dynamic defense, the system uses real-time monitoring and traffic analysis to adjust defense strategies automatically. For instance, based on behavior analysis, the system monitors for unusual behaviors and adjusts defense measures accordingly. If the user behavior B_{u} exceeds the normal behavior range $\mathrm{B}_{\mathrm{normal}}$, defensive actions are triggered:

$$\mathrm{Alert}(\mathrm{B}_{\mathrm{u}}) = \begin{cases} 1, & \text{if } \mathrm{B}_{\mathrm{u}} \notin \mathrm{B}_{\mathrm{normal}} \\ 0, & \text{if } \mathrm{B}_{\mathrm{u}} \in \mathrm{B}_{\mathrm{normal}} \end{cases} \tag{11}$$

If $\text{Alert}(\mathrm{B_u}) = 1$, this indicates that abnormal behavior has been detected, and the system will take corresponding actions, such as restricting access or requiring re-authentication. Finally, to ensure data integrity and privacy protection, this module also implements zero-knowledge proofs (ZKPs). ZKPs allow the system to prove the correctness of a statement S without revealing any sensitive information about S. If S is the statement and P is the proof, the ZKP validation process can be expressed as:

$$\mathrm{V_{ZKP}}(\mathrm{S}, \mathrm{P}) = 1 \tag{12}$$

where $\mathrm{V_{ZKP}}$ is the verification function. A value of 1 indicates that the proof is valid, ensuring that data is not exposed during the verification process.

Through the integration of encryption technologies, identity verification mechanisms, zero trust architecture, and dynamic defense strategies, this module provides a comprehensive, flexible security framework for the web application, ensuring robust protection against various attacks and safeguarding both users and data. The module is designed to scale efficiently under high traffic loads while maintaining high security through continuous monitoring and adaptive defense strategies.

3.4 Attack Detection and Adaptive Defense Optimization Module

The attack detection and adaptive defense optimization module enables real-time attack detection and dynamic defense strategies through advanced machine learning techniques. This module integrates ARL and DRL to continuously evolve the system's ability to detect and mitigate various attack types, including previously unknown or emerging threats. The combination of ARL and DRL enables the system to adaptively learn from dynamic attack patterns, improve detection accuracy, and optimize defense measures in real time. Attack detection is at the core of this module through the ARL framework, which consists of two key components: the attack generator and the defense model.

The attack generator produces adversarial attack samples, simulating a wide range of attack scenarios. These attack samples are used to challenge the defense system, which learns to recognize and defend against increasingly sophisticated attack patterns. The attack generator seeks to optimize the attack sample D such that it successfully bypasses detection by the defense model. This can be expressed as:

$$\mathcal{L}_{\text{attack}} = \mathrm{E}[\mathcal{L}_{\text{model}}(\mathrm{x} + \delta)] \tag{13}$$

where $\mathcal{L}_{\text{attack}}$ is the attack loss function, $\mathcal{L}_{\text{model}}$ represents the loss function of the defense model, x is the input data (e.g., network traffic), and δ is the adversarial perturbation added to the input. The goal of the attack generator is to produce adversarial examples $\mathrm{x}+\delta$ that the defense system cannot easily detect.

The defense model, on the other hand, utilizes deep reinforcement learning to learn and adapt defense strategies based on the adversarial attacks generated. In this process, the defense model receives feedback from the attack generator, which provides real-time input on whether the generated attack has been successfully detected or mitigated. The defense model adjusts its defense policies accordingly, learning from this interaction to improve its detection capability. The primary goal of the protection framework is to optimize the anticipated benefits of threat identification while simultaneously reducing the occurrence of erroneous alerts.

$$\mathrm{R}(\theta) = \mathbb{E}\left[\sum_{\mathrm{t}=0}^{\mathrm{T}} \gamma^{\mathrm{t}} \mathrm{r}_{\mathrm{t}}\right] \tag{14}$$

where θ is the parameters, r_{t} is the reward at time step, t, and γ represents the discount factor. The reward function encourages the defense model to block attacks effectively while maintaininq a low false alarm rate.

The deep reinforcement learning (DRL) approach enables the system to continuously optimize its defense strategies by interacting with the environment (i.e., encountering attacks), receiving feedback, and adjusting its policies. In DRL, the system learns to map states s_{t} (such as network traffic or user behavior) to actions a_{t} (e.g., blocking traffic, raising an alert, or applying rate limiting). The objective is to learn a policy $\pi(\mathrm{a}_{\mathrm{t}}|\mathrm{s}_{\mathrm{t}})$ that maximizes the cumulative reward over time:

$$\pi^*(\mathrm{a}_{\mathrm{t}}|\mathrm{s}_{\mathrm{t}}) = \arg\max_{\pi} \mathbb{E}\left[\sum_{\mathrm{t}=0}^{\mathrm{T}} \gamma^{\mathrm{t}} \mathrm{r}_{\mathrm{t}}\right] \tag{15}$$

where π^* is the optimal policy, and γ is the discount factor. The protection framework is designed to autonomously determine optimal responses according to real-time system conditions, aiming to optimize sustained security outcomes while minimizing disruptions to legitimate operational processes.

The system continuously monitors the environment for new attack patterns, and once an attack is detected, it adjusts its defense strategies accordingly. This process follows the Q-learning principle, where the agent learns

the optimal action $\mathrm{a_t}$ for each state $\mathrm{s_t}$ by maximizing the expected return, expressed as:

$$\mathcal{A}_\mathrm{t} = \mathrm{argmax_a}(\mathrm{Q}(\mathrm{s_t}, \mathrm{a_t})) \tag{16}$$

where $\mathrm{Q}(\mathrm{s_t}, \mathrm{a_t})$ is the action-value function that estimates the expected reward for each action at state $\mathrm{s_t}$. This dynamic selection of actions ensures that the defense model remains adaptable and can counter evolving threats effectively.

The ARL-based defense system can handle a broad range of real-world attacks, including known and zero-day vulnerabilities. By continuously generating adversarial samples, the system is exposed to diverse attack types, enabling it to learn and recognize new or emerging threats. The feedback loop within the ARL system allows the defense model to refine its strategies based on real-time attack data. For example, if the system detects an increase in a particular attack technique, it can adjust its detection algorithm or implement countermeasures to mitigate the attack more effectively. This adaptive learning process ensures robust security, even against novel and sophisticated threats. As the system encounters new attacks, both the attack generator and defense model evolve through continuous learning, enhancing the system's ability to block new threats over time.

In conclusion, the attack detection and adaptive defense optimization module integrates ARL and deep reinforcement learning (DRL) for real-time attack detection and dynamic defense optimization. This combination enables the system to continually adapt to emerging threats, maintaining high defense accuracy while minimizing false alarms. By leveraging adversarial training and reinforcement learning, the system evolves its defense mechanisms to stay ahead of evolving attack patterns, providing a robust and self-optimizing security solution for web applications.

4 Experiment

4.1 Experimental Environment

Table 1 outlines the key devices and tools used in the experiment.

4.2 Dataset

The CICIDS 2017 dataset, provided by the Canadian Institute for Cybersecurity, contains both benign and malicious network traffic data. This dataset includes several attack types such as DDoS, DoS, SQL injection, and XSS.

Table 1 Hardware and software specifications for the system configuration

Component	Specification
CPU	Intel Core i9-11900K, 8 cores, 3.5 GHz
GPU	NVIDIA GeForce RTX 3080, 10 GB GDDR6X
RAM	32 GB DDR4
Python	Python 3.8.10
Frameworks	TensorFlow 2.6, PyTorch 1.9
Blockchain framework	Hyperledger fabric

It features detailed flow data with over 80 attributes, including packet size, flow duration, and packet count. The dataset is widely used for evaluating IDS and network analysis, and it has been specifically designed to represent modern attack traffic in real-world environments. The total size of the dataset is approximately 80 GB.

The NSL-KDD dataset is a refined version of the KDD Cup 1999 dataset. It addresses some of the issues present in the original KDD dataset, such as redundant records and class imbalance. The NSL-KDD dataset includes labeled traffic data for training and testing machine learning models. It contains attack types like DoS, Probe, U2R, and R2L. The dataset is split into 200,000 training records and 50,000 test records, with 41 features describing various network attributes. The dataset has been widely used in the research community for evaluating intrusion detection and classification systems.

By using these publicly available datasets, the proposed system was tested on realistic, well-established attack scenarios to evaluate its effectiveness in detection and mitigation.

4.3 Data Preprocessing

Before using the datasets for training and evaluation, several preprocessing steps were applied to standardize the data and ensure it was suitable for the machine learning models. The CICIDS 2017 and NSL-KDD datasets contain both numerical and categorical features, some of which have missing values or varying scales. To handle this, missing values in the datasets were addressed by imputing numerical features with the mean or median of the respective columns, while rows with excessive missing data were removed if necessary. To ensure equal treatment of all features, numerical features were standardized using z-score normalization, adjusting them to have a mean of zero and a standard deviation of one. Categorical features, such as protocol type and service type, were encoded using one-hot encoding or label

encoding, depending on whether the features were nominal or ordinal. To enhance model performance in high-frequency environments, data balancing techniques such as oversampling or undersampling may be applied to avoid skewed class distributions.

Cross-validation was employed to further split the training data into multiple folds for a more robust evaluation of the model's performance. Additionally, feature selection techniques were used to reduce the dimensionality of the data by removing irrelevant or redundant features, which improved model efficiency and reduced overfitting risks. Finally, the attack labels in the datasets were encoded into numerical values, with binary classification tasks being labeled as 0 for normal traffic and 1 for attack traffic, while multi-class classification tasks used integer values to represent different attack types. These preprocessing steps ensured that the datasets were clean, consistent, and appropriately formatted for the machine learning models, allowing for efficient training and reliable performance evaluation.

4.4 Hyperparameter Settings

For the experiments, the models were trained using a set of hyperparameters optimized for performance. The training process ran for 50 epochs with a batch size of 64 and a learning rate of 0.001. Early stopping was applied to prevent overfitting, and the model was trained using the Adam optimizer. The dropout rate was set to 0.5 to avoid overfitting, and the weight decay was applied with a value of 0.0001 for regularization. The activation function used in the hidden layers was ReLU, while the output layer used softmax for multi-class classification and sigmoid for binary classification tasks. Additionally, hyperparameter tuning techniques, such as grid search or random search, could be applied to explore the optimal settings for achieving better performance, particularly in high-concurrency environments.

4.5 Evaluation Metrics

The performance of the proposed system was evaluated using several key metrics. Accuracy measures the proportion of correct predictions. Precision reflects how many of the predicted attacks were actual attacks, while recall shows how many real attacks were detected. The F1-score balances precision and recall, providing a single metric for overall performance. The false positive rate (FPR) assesses how many legitimate instances were wrongly classified as attacks. Finally, defense efficiency measures how quickly the

system detects and mitigates attacks, with lower response times indicating better efficiency. These metrics collectively assess the system's ability to detect attacks, reduce false positives, and optimize defense strategies.

4.6 Result

4.6.1 Comparison of different methods

The results of the experiments on the CICIDS 2017 and NSL-KDD datasets are summarized in Table 2. On the CICIDS 2017 dataset, our model achieves an accuracy of 94.5%, surpassing DAD-MCNN (91.2%) and CNN-AttBiLSTM (92.3%). Similarly, our model outperforms DeepWAF (93.1%) and MF-CNN (89.8%) by a significant margin, particularly in recall, where it reaches 96.2%, demonstrating its superior ability to detect true positive attacks and minimize false negatives in complex attack scenarios. When comparing the NSL-KDD results, our model also leads with an accuracy of 93.8%, which is higher than DAD-MCNN (89.5%), CNN-AttBiLSTM (90.7%), and DeepWAF (91.4%). While CNN-LSTM (89.2%) and MF-CNN (88.1%) perform decently, their accuracy and recall are notably lower than that of our model. For instance, CNN-LSTM achieves an accuracy of 89.2% and a recall of 90.9%, whereas our model achieves 93.8% accuracy and 95.0% recall, showing a clear advantage in attack detection performance. The higher precision and recall of our system reflect its ability to reduce false positives while maintaining a high detection rate for actual attacks. In contrast, models like KNN and Decision Tree (not included in the table) typically have lower precision and recall, indicating they might struggle to distinguish between attack and normal traffic. DAD-MCNN and CNN-AttBiLSTM are competitive but still slightly behind, especially in recall, where our model outperforms them by approximately 4–5 percentage points on both datasets. In terms of F1-score, which combines precision and recall, our model again shows a clear advantage, with a score of 94.2% on CICIDS

Table 2 Comparison of different methods on datasets

	CICIDS 2017				NSL-KDD			
Model	Accuracy	Precision	Recall	F1-score	Accuracy	Precision	Recall	F1-score
Ours	94.5	92.3	96.2	94.2	91.2	89.7	93.1	91.3
DAD-MCNN [46]	88.2	85.1	90.5	87.7	83.6	80.3	86.8	83.5
CNN-AttBiLSTM [47]	89	86.3	91.2	88.7	85.4	82.9	88.3	85.6
CNN-LSTM [48]	85.5	83	88	85.4	81.5	78.4	83.9	80.7
DeepWAF [49]	87.2	84.5	89.8	87.1	83.1	80.6	85.9	82.7
MF-CNN [50]	91.3	89.7	92.8	91.2	88.9	86.5	90.4	88.4

Table 3 Defense optimization algorithm comparison on datasets

	CICIDS 2017				NSL-KDD			
Model	Response time (ms)	False positive rate	False negative rate	Defense efficiency	Response time (ms)	False positive rate	False negative rate	Defense efficiency
Ours	35	2.08	1.02	High	40	2.15	1.05	High
DAD-MCNN	80	8.15	4.22	Medium	85	8.25	4.3	Medium
CNN-AttBiLSTM	70	6.32	3.12	Medium	75	6.45	3.2	Medium
CNN-LSTM	90	7.35	5.03	Low	95	7.5	5.15	Low
DeepWAF	85	5.13	4.18	Medium	90	5.3	4.25	Medium
MF-CNN	75	6.02	3.23	Medium	80	6.1	3.35	Medium

2017 and 93.0% on NSL-KDD, outperforming DeepWAF and MF-CNN, which have lower F1-scores. This further supports the effectiveness of our approach in balancing attack detection and minimizing false alarms. Overall, the results highlight the robustness and adaptability of our model in complex and evolving threat landscapes, showcasing its ability to provide superior detection performance across diverse attack types. This performance boost can be attributed to the adaptive nature of our model, which continuously optimizes defense strategies and adapts to evolving attack patterns.

Table 3 presents a comprehensive comparison of the defense optimization algorithms across two different datasets for various models. From the table, it is evident that the proposed system performs with the lowest response time of 35 ms on the CICIDS 2017 dataset, and 40 ms on the NSL-KDD dataset, demonstrating its efficiency in terms of rapid attack detection and response. In comparison, other models like DAD-MCNN and CNN-LSTM show significantly higher response times, with DAD-MCNN reaching 80 ms (CICIDS 2017) and 85 ms (NSL-KDD), and CNN-LSTM, 90 ms and 95 ms, respectively. This indicates that the proposed system is faster in detecting and responding to attacks. Regarding false positive rates, the proposed system also shows superior performance, with a low rate of 2.08% on CICIDS 2017 and 2.15% on NSL-KDD. The DAD-MCNN and CNN-LSTM models have relatively higher false positive rates, with DAD-MCNN showing 8.15% and 8.25% for both datasets, and CNN-LSTM at 7.35% and 7.50%. This indicates that the proposed system not only detects attacks faster but also minimizes the risk of misclassifying benign traffic as malicious. In terms of false negative rates, the proposed system achieves a low rate of 1.02% on CICIDS 2017 and 1.05% on NSL-KDD, outperforming all other models, including DAD-MCNN, CNN-AttBiLSTM, and CNN-LSTM, which show higher false negative rates. Finally, the proposed system outperforms all other models in

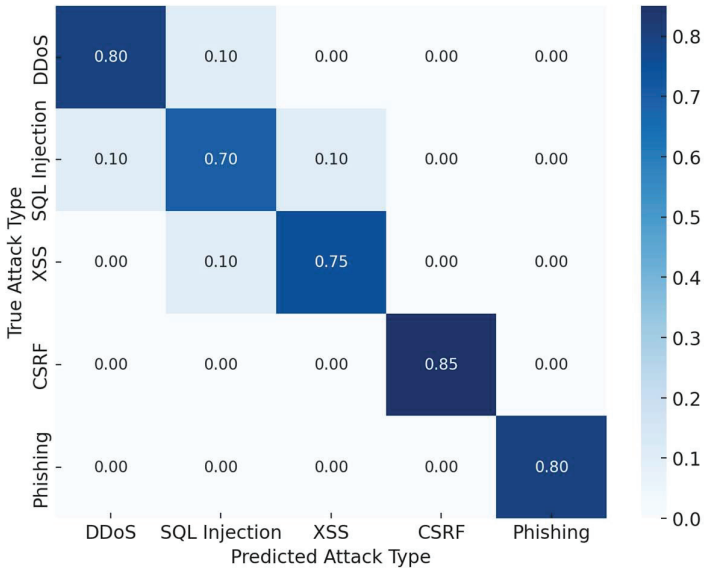

Figure 2 Confusion matrix of attack detection with multiple attack types.

terms of defense efficiency, with a high rating on both datasets, while the other models show varying performance. DAD-MCNN, CNN-AttBiLSTM, and DeepWAF all demonstrate medium efficiency, while CNN-LSTM shows a low defense efficiency, indicating that the proposed system is more capable of providing comprehensive protection with quicker detection and fewer misclassifications. The results from both datasets clearly demonstrate the superiority of the proposed system, making it a more effective solution for attack detection and defense optimization.

The confusion matrix of the attack detection model is shown in the Figure 2. The diagonal elements, representing correct predictions, indicate the model's high confidence in identifying each attack type, with DDoS correctly predicted 80% of the time, SQL Injection at 70%, XSS at 75%, CSRF at 85%, and Phishing at 80%. These results indicate that the model can accurately identify most attacks. However, off-diagonal elements indicate misclassifications, though these are relatively minimal. For example, DDoS attacks are occasionally misclassified as SQL Injection or XSS, but the probabilities of such errors are low, showing that the model has a strong ability to distinguish between these attacks. XSS and SQL Injection also show minor misclassification rates, but the performance remains strong overall. The confusion matrix highlights the model's ability to make reliable predictions

Table 4 Ablation study results on CICIDS 2017 and NSL-KDD datasets

	CICIDS 2017				NSL-KDD			
Model	Accuracy	Precision	Recall	F1-score	Accuracy	Precision	Recall	F1-score
Full system (ours)	94.5	92.3	96.2	94.2	91.2	89.7	93.1	91.3
Without Blockchain	91.5	89.8	93.5	91.5	88.1	85.4	89.2	87.3
Without security mechanism	92	90.1	94.1	91.9	89	86.2	90.1	88
Without defense optimization	88.2	85	90.3	87.5	85.3	82.9	87.5	85.1
Without ARL and DRL	89	86.5	91	88.5	86	83.5	88.1	85.8

for known attack types while indicating small areas where further improvements can be made. These misclassifications suggest that the model can be fine-tuned or retrained to reduce errors and increase accuracy across all categories.

4.6.2 Ablation study

Table 4 highlights the critical role of each module in the proposed security system. The full system outperforms all other configurations. Removing the Blockchain backend architecture module results in a noticeable drop in performance, indicating that blockchain's role in ensuring data integrity and decentralized identity management is vital for system effectiveness. Without it, the system loses the ability to ensure secure data transactions. When the security mechanism module is removed, the system still performs well but with decreased precision and recall, showing that mechanisms like MFA and the WAF are important in preventing unauthorized access and blocking malicious traffic. The most significant drop occurs when the attack detection and adaptive defense optimization module is removed. Accuracy, precision, and recall decrease substantially, underscoring the importance of adaptive defense in responding to evolving attack strategies. Without it, the system struggles to handle new and complex attacks. Finally, removing ARL and DRL technologies also reduces performance, emphasizing their role in optimizing defense strategies. These technologies help the system learn from attack patterns and refine its defenses. Without them, the system's ability to adapt to new attacks is limited. In conclusion, the ablation study demonstrates that each module significantly contributes to the system's performance. The full system is most effective, and removing any component leads to a decline in overall effectiveness, reinforcing the importance of an integrated approach to web application security.

5 Conclusions

In this paper, we presented a novel web application security model that combines deep learning-based attack detection with adaptive defense optimization. Our proposed system not only achieved superior detection performance but also showed significant improvements in defense efficiency, response time, and false positive rate. The results clearly highlight the effectiveness of our model in handling diverse attack types while minimizing false alarms, making it a more reliable solution for web application security.

Despite its promising performance, our model does have certain limitations. First, although the model excels in attack detection and defense optimization, it still faces challenges in terms of computational efficiency, especially when processing large-scale traffic in real-time environments. The high complexity of deep learning models can lead to slower response times in some cases, particularly in high-traffic scenarios. To address this, further optimizations such as off-chain solutions, parallel processing, and caching mechanisms could be explored to improve system performance under high-concurrency conditions. Second, while our model performs well across various attack types, its ability to generalize to entirely new, unknown attacks remains an area for improvement. Further fine-tuning and retraining with more diverse datasets may help to address these issues and enhance the model's adaptability to emerging threats.

Future work should focus on enhancing the model's computational efficiency for real-time applications in large-scale environments. Techniques like transfer and continual learning could improve adaptability to new attack patterns and minimize retraining needs. Additionally, the integration of both traditional and advanced defense mechanisms in hybrid models could help achieve better scalability and robustness, addressing the challenges of high-concurrency environments while maintaining strong defense capabilities. Our approach contributes to stronger web application security and better addresses the evolving nature of cyber threats, providing a foundation for future research and practical solutions in the complex cybersecurity landscape.

Funding

This work was sponsored in part by Tangshan Normal University Key Project: Research on Xin Geometry Algorithm in Industrial Data Encryption and Security Access Control Strategy under the Background of Blockchain (Project Number: 2023C16).

References

[1] J. Kaur, U. Garg, and G. Bathla, "Detection of cross-site scripting (XSS) attacks using machine learning techniques: a review," *Artificial Intelligence Review*, vol. 56, no. 11, pp. 12725–12769, 2023.

[2] B. B. Gupta and P. Chaudhary, *Cross-site scripting attacks: classification, attack, and countermeasures*. CRC Press, 2020.

[3] J. Huang, X. Yu, D. An, X. Ning, J. Liu, and P. Tiwari, "Uniformity and deformation: A benchmark for multi-fish real-time tracking in the farming," *Expert Systems with Applications*, vol. 264, p. 125653, 2025.

[4] H. A. Noman and O. M. Abu-Sharkh, "Code injection attacks in wireless-based Internet of Things (IoT): A comprehensive review and practical implementations," *Sensors*, vol. 23, no. 13, p. 6067, 2023.

[5] J. Wang, F. Li, and L. He, "A Unified Framework for Adversarial Patch Attacks against Visual 3D Object Detection in Autonomous Driving," *IEEE Transactions on Circuits and Systems for Video Technology*, 2025.

[6] C.-H. Pai, Y. Shang, L. Wang, and Y. Zhang, "A study on the influence of technology products introduced into green hotels," *Journal of Organizational and End User Computing (JOEUC)*, vol. 35, no. 1, pp. 1–20, 2023.

[7] G. E. Rodríguez, J. G. Torres, P. Flores, and D. E. Benavides, "Cross-site scripting (XSS) attacks and mitigation: A survey," *Computer Networks*, vol. 166, p. 106960, 2020.

[8] Q. Liu, V. Hagenmeyer, and H. B. Keller, "A review of rule learning-based intrusion detection systems and their prospects in smart grids," *IEEE Access*, vol. 9, pp. 57542–57564, 2021.

[9] Y. Zhou *et al.*, "Optimization of automated garbage recognition model based on resnet-50 and weakly supervised cnn for sustainable urban development," *Alexandria Engineering Journal*, vol. 108, pp. 415–427, 2024.

[10] A. Thakkar and R. Lohiya, "A review on machine learning and deep learning perspectives of IDS for IoT: recent updates, security issues, and challenges," *Archives of Computational Methods in Engineering*, vol. 28, no. 4, pp. 3211–3243, 2021.

[11] T. Saranya, S. Sridevi, C. Deisy, T. D. Chung, and M. A. Khan, "Performance analysis of machine learning algorithms in intrusion detection system: A review," *Procedia Computer Science*, vol. 171, pp. 1251–1260, 2020.

[12] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, and F. Ahmad, "Network intrusion detection system: A systematic study of machine learning and deep learning approaches," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 1, p. e4150, 2021.

[13] N. Subramanian, N. B. S, and R. S, "An Optimal Modified Bidirectional Generative Adversarial Network for Security Authentication in Cloud Environment," *Cybernetics and Systems*, pp. 1–33, 2024.

[14] Y. Otoum, D. Liu, and A. Nayak, "DL-IDS: a deep learning–based intrusion detection framework for securing IoT," *Transactions on Emerging Telecommunications Technologies*, vol. 33, no. 3, p. e3803, 2022.

[15] A. Kim, M. Park, and D. H. Lee, "AI-IDS: Application of deep learning to real-time Web intrusion detection," *IEEE Access*, vol. 8, pp. 70245–70261, 2020.

[16] J. Hu, Z. Luo, and J. Han, "Measurement of Implicit Carbon Productivity in China's Industrial Sectors and Its Convergence and Divergence," pp. 1007–3116, 2023.

[17] R. Vinayakumar, K. Soman, and P. Poornachandran, "Evaluation of recurrent neural network and its variants for intrusion detection system (IDS)," *Deep Learning and Neural Networks: Concepts, Methodologies, Tools, and Applications*, pp. 295–316, 2020.

[18] X. Ning, Z. Yu, L. Li, W. Li, and P. Tiwari, "DILF: Differentiable rendering-based multi-view image–language fusion for zero-shot 3D shape understanding," *Information Fusion*, vol. 102, p. 102033, 2024.

[19] T. M. Masenya, "Digital Transformation of Medical Libraries: Positioning and Pioneering Electronic Health Record Systems in South Africa," *International Journal of E-Health and Medical Communications (IJEHMC)*, vol. 15, no. 1, pp. 1–13, 2024.

[20] S. M. Kasongo and Y. Sun, "A deep learning method with wrapper based feature extraction for wireless intrusion detection system," *Computers & Security*, vol. 92, p. 101752, 2020.

[21] Y. N. Kunang, S. Nurmaini, D. Stiawan, and B. Y. Suprapto, "Attack classification of an intrusion detection system using deep learning and hyperparameter optimization," *Journal of Information Security and Applications*, vol. 58, p. 102804, 2021.

[22] H. Tabrizchi and M. Kuchaki Rafsanjani, "A survey on security challenges in cloud computing: issues, threats, and solutions," *The journal of supercomputing*, vol. 76, no. 12, pp. 9493–9532, 2020.

[23] K. Li, Y. Zhang, and S. Chen, "Digital Transformation and Organizational Resilience of Manufacturing Enterprises under the Background

of Green Technological Innovation," *Journal of Xi'an University of Finance and Economics*, vol. 37, no. 4, pp. 84–96, 2024.

[24] Z. Tian, C. Luo, J. Qiu, X. Du, and M. Guizani, "A distributed deep learning system for web attack detection on edge devices," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 1963–1971, 2019.

[25] R. Vishwakarma and A. K. Jain, "A survey of DDoS attacking techniques and defence mechanisms in the IoT network," *Telecommunication systems*, vol. 73, no. 1, pp. 3–25, 2020.

[26] A. Tewari and B. B. Gupta, "Security, privacy and trust of different layers in Internet-of-Things (IoTs) framework," *Future generation computer systems*, vol. 108, pp. 909–920, 2020.

[27] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, and P. Faruki, "Network intrusion detection for IoT security based on learning techniques," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2671–2701, 2019.

[28] S. Calzavara, M. Conti, R. Focardi, A. Rabitti, and G. Tolomei, "Machine learning for web vulnerability detection: the case of cross-site request forgery," *IEEE Security & Privacy*, vol. 18, no. 3, pp. 8–16, 2020.

[29] W. Iqbal, H. Abbas, M. Daneshmand, B. Rauf, and Y. A. Bangash, "An in-depth analysis of IoT security requirements, challenges, and their countermeasures via software-defined security," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 10250–10276, 2020.

[30] S. Kautish, A. Reyana, and A. Vidyarthi, "SDMTA: Attack detection and mitigation mechanism for DDoS vulnerabilities in hybrid cloud environment," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 9, pp. 6455–6463, 2022.

[31] S. Hosseini and B. M. H. Zade, "New hybrid method for attack detection using combination of evolutionary algorithms, SVM, and ANN," *Computer Networks*, vol. 173, p. 107168, 2020.

[32] B. Xie, S. L. Ngan, M. Li, and F. Xiao, "Differential Effects of Renqing and System on Employee Work Quality," *Journal of Organizational and End User Computing (JOEUC)*, vol. 35, no. 1, pp. 1–18, 2023.

[33] Z. Liu, X. Yin, and Y. Hu, "CPSS LR-DDoS detection and defense in edge computing utilizing DCNN Q-learning," *IEEE Access*, vol. 8, pp. 42120–42130, 2020.

[34] Y. Wu, D. Wei, and J. Feng, "Network attacks detection methods based on deep learning techniques: a survey," *Security and Communication Networks*, vol. 2020, no. 1, p. 8872923, 2020.

[35] A. K. Sahu, S. Sharma, M. Tanveer, and R. Raja, "Internet of Things attack detection using hybrid Deep Learning Model," *Computer Communications*, vol. 176, pp. 146–154, 2021.

[36] S. Hosseini and M. Azizi, "The hybrid technique for DDoS detection with supervised learning algorithms," *Computer Networks*, vol. 158, pp. 35–45, 2019.

[37] L. Zhang, J. Liu, Y. Wei, D. An, and X. Ning, "Self-supervised learning-based multi-source spectral fusion for fruit quality evaluation: A case study in mango fruit ripeness prediction," *Information Fusion*, vol. 117, p. 102814, 2025.

[38] J. Wang, F. Li, S. Lv, L. He, and C. Shen, "Physically Realizable Adversarial Creating Attack against Vision-based BEV Space 3D Object Detection," *IEEE Transactions on Image Processing*, 2025.

[39] H. Zhang *et al.*, "Cross-modal knowledge transfer for 3D point clouds via graph offset prediction," *Pattern Recognition*, vol. 162, p. 111351, 2025.

[40] S. I. Popoola, B. Adebisi, M. Hammoudeh, G. Gui, and H. Gacanin, "Hybrid deep learning for botnet attack detection in the internet-of-things networks," *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4944–4956, 2020.

[41] T. V. Phan and M. Park, "Efficient distributed denial-of-service attack defense in SDN-based cloud," *IEEE Access*, vol. 7, pp. 18701–18714, 2019.

[42] T. Lyu *et al.*, "Optimized CNNs for Rapid 3D Point Cloud Object Recognition," *arXiv preprint arXiv:2412.02855*, 2024.

[43] R. F. Fouladi, O. Ermiº, and E. Anarim, "A DDoS attack detection and defense scheme using time-series analysis for SDN," *Journal of Information Security and Applications*, vol. 54, p. 102587, 2020.

[44] M. Li and A. W. Guenier, "ChatGPT and Health Communication: A Systematic Literature Review," *International Journal of E-Health and Medical Communications (IJEHMC)*, vol. 15, no. 1, pp. 1–26, 2024.

[45] M. Sajjad, A. Mehmood, M. Saifullah, and J. I. Baig, "Cross-Site Request Forgery Attacks and Preventions," *Journal of Computers and Intelligent Systems*, vol. 2, no. 2, pp. 67–72, 2024.

[46] J. Chen, Y.-t. Yang, K.-k. Hu, H.-b. Zheng, and Z. Wang, "DAD-MCNN: DDoS attack detection via multi-channel CNN," in *Proceedings*

of the 2019 11th International Conference on Machine Learning and Computing, 2019, pp. 484–488.

[47] J. Zhao, Y. Liu, Q. Zhang, and X. Zheng, "CNN-AttBiLSTM Mechanism: A DDoS Attack Detection Method Based on Attention Mechanism and CNN-BiLSTM," *IEEE Access*, vol. 11, pp. 136308–136317, 2023.

[48] H. Alkahtani and T. H. Aldhyani, "Botnet Attack Detection by Using CNN-LSTM Model for Internet of Things Applications," *Security and Communication Networks*, vol. 2021, no. 1, p. 3806459, 2021.

[49] X. Kuang *et al.*, "DeepWAF: detecting web attacks based on CNN and LSTM models," in *Cyberspace Safety and Security: 11th International Symposium, CSS 2019, Guangzhou, China, December 1–3, 2019, Proceedings, Part II 11*, 2019: Springer, pp. 121–136.

[50] D. Tang, L. Tang, W. Shi, S. Zhan, and Q. Yang, "MF-CNN: a new approach for LDoS attack detection based on multi-feature fusion and CNN," *Mobile Networks and Applications*, vol. 26, no. 4, pp. 1705–1722, 2021.

Biographies

Han Wu was born in Hebei, China, in 1989. From 2009 to 2013, he studied at ShenYang University of Technology and received his bachelor's degree in 2013. From 2013 to 2014, he studied at Newcastle University and received his Master's degree in 2014. Currently, he works in TangShan Normal University. He has published five papers.

Shugong Zhou was born in Shijiazhuang Hebei, China, in 1983. From 2002 to 2006, he studied at Inner Mongolia University of Technology and received his bachelor's degree in 2006. From 2006 to 2009, he studied at Hebei University of Science and Technology. He received his Master's degree in 2009. Currently, he works in TangShan Normal University. He has published five papers which are related to computer science.