REAL-TIME SQL INJECTION ATTACK DETECTION IN NETWORK ENVIRONMENTS

A PROJECT REPORT

Submitted by

ROSHAN KUMAR	**810421104142**
RUPESH KUMAR	**810421104143**
VIKASH KUMAR	**810421104188**

In partial fulfillment for the award of degree

of

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE AND ENGINEERING

DHANALAKSHMI SRINIVASAN ENGINEERING COLLEGE

(AUTONOMOUS)

PERAMBALUR - 621 212

ANNA UNIVERSITY : CHENNAI 600 025

MAY 2025

DHANALAKSHMI SRINIVASAN ENGINEERING COLLEGE

(AUTONOMOUS)

PERAMBALUR – 621 212

BONAFIDE CERTIFICATE

Certified that this project report **"REAL-TIME SQL INJECTION ATTACK DETECTION IN NETWORK ENVIRONMENTS"** is the bonafide work of **"ROSHAN KUMAR (810421104142), RUPESH KUMAR (810421104143), VIKASH KUMAR (810421104188)"** who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Dr. R. GOPI, M.Tech., Ph.D., (PDF).,	**Mrs. M. HEMALATHA, ME.,**
PROFESSOR And HEAD,	**SUPERVISOR**
Department of Computer Science and Engineering,	Department of Computer Science and Engineering,
Dhanalakshmi Srinivasan Engineering College (Autonomous),	Dhanalakshmi Srinivasan Engineering College (Autonomous),
Perambalur – 621 212.	Perambalur – 621 212.

Submitted for Project Viva-Voce Examination held on _______________

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We express our gratitude and thanks to **Our Parents** first for giving health and a sound mind for completing this mini project. We give all the glory and thanks to our almighty **GOD** for showering upon, the necessary wisdom and grace for accomplishing this project.

It is our pleasant duty to express a deep sense of gratitude to our honourable Chancellor, **Shri. A. Srinivasan**, for his kind encouragement. We sincerely thank our principal **Dr. D. Shanmugasundram, M.E., Ph.D., F.I.E., C.Eng.,** our DEAN **Dr. K. Anbarasan M.E., Ph.D.**, and our COE, **Dr. M. Chellapan, M.E., Ph.D.,** for their unflinching devotion, which enabled us to complete this project.

We express our faithful and sincere gratitude to our Head of the Department **Dr. R. Gopi, M.Tech., Ph.D., (PDF)** for his valuable guidance and support that he gave us during the project time.

We express our faithful and sincere gratitude to our Project Coordinator **Mrs. B. Deepika, M.E., (Ph.D).,** of Department of Computer Science and Engineering for giving support throughout our project.

We also thankful to our internal guide **Mrs. M. Hemalatha, M.E.,** of Department of Computer Science and Engineering for her valuable guidance and precious suggestion to complete this project work successfully.

We render our thanks to all **Faculty members** and **Programmers** of Department of **Computer Science and Engineering** for their timely assistance.

DHANALAKSHMI SRINIVASAN ENGINEERING COLLEGE

(AUTONOMOUS)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Vision and Mission of the Department:

Vision

To produce globally competent, socially responsible professionals in the field of Computer Science and Engineering.

Mission

M1: Impart high quality experiential learning to get expertise in modern

M2: Inculcate industry exposure and build inter disciplinary research skills

M3: Mould the students to become Software Professionals, Researchers and Entrepreneurs by providing advanced laboratories.

M4: Acquire Innovative skills and promote lifelong learning with a sense of societal and ethical responsibilities

Program Educational Objectives (PEOs)

PEO 1: Graduates of the programme will develop proficiency in identifying, formulating, and resolving complex computing problems.

PEO 2: Graduates of the programme will achieve successful careers in the field of computer science and engineering, pursue advanced degrees, or demonstrate entrepreneurial success.

PEO 3: Graduates of the programme will cultivate effective communication skills, teamwork abilities, ethical values, and leadership qualities for professional engagement in industry and research organizations.

ABSTRACT

SQL injection (SQLi) remains a critical threat to web application security, enabling attackers to manipulate backend databases through malicious input. This project proposes the development of a **SQL Injection Detection Network (SIDN)** aimed at identifying and mitigating SQLi attacks in real-time. The system utilizes machine learning techniques to analyze and classify user input based on patterns associated with both legitimate and malicious queries. By training on a comprehensive dataset, the model is capable of detecting known and novel SQLi payloads with high accuracy. The architecture is designed to integrate seamlessly with existing web applications, offering a lightweight yet effective layer of security. The proposed solution enhances cybersecurity by providing adaptive protection against evolving attack vectors, thereby reducing the risk of data breaches and unauthorized access.

As digital systems become increasingly integrated into every aspect of modern life, the importance of robust cybersecurity measures has never been greater. This project focuses on enhancing cybersecurity by developing intelligent and adaptive defense mechanisms capable of detecting and mitigating potential threats in real-time. Leveraging advanced technologies such as machine learning, anomaly detection, and behavioral analysis, the system aims to identify suspicious activities and prevent unauthorized access to sensitive data. The proposed approach not only improves threat detection accuracy but also reduces response time and minimizes false positives. By continuously learning from new attack patterns and adapting to evolving threats, this project contributes to building a more secure and resilient digital infrastructure suitable for today's dynamic cyber landscape.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

YOLO	-	You Only Look Once
OPENCV	-	Open Computer Vision
HOG	-	Histogram Of Oriented Gradients
SVM	-	Support Vector Machine
SSD	-	Single Shot Multibox Detector
ADAS	-	Advanced Driver Assistance System
R-CNN	-	Region Based Convolutional Neural Network
ANN	-	Artificial Neural Network
CNN	-	Convolutional Neural Network
COCO	-	Common Objects In Context

CHAPTER 1

INTRODUCTION

1.1 CYBERSECURITY

Cybersecurity refers to the practice of protecting computers, servers, mobile devices, networks, and data from malicious attacks, unauthorized access, damage, or theft. It encompasses a wide range of technologies, processes, and practices designed to safeguard digital systems and ensure the confidentiality, integrity, and availability of information.

Cybersecurity refers to the practice of protecting computer systems, networks, and data from unauthorized access, damage, or theft. It encompasses a wide range of technologies, processes, and practices designed to safeguard digital assets against evolving cyber threats. As attackers become more sophisticated, traditional security measures such as firewalls and antivirus software are no longer sufficient. Modern cybersecurity strategies now integrate advanced technologies such as artificial intelligence (AI), machine learning (ML)proactively detect and respond to potential risks.

This project aims to contribute to the field of cybersecurity by exploring and implementing intelligent solutions that enhance system resilience and reduce the risk of cyberattacks. By focusing on real-time detection and prevention mechanisms, the goal is to create adaptive, efficient, and scalable security solutions suitable for today's dynamic threat environment.

Cybersecurity is a rapidly evolving field focused on protecting digital systems, networks, devices, and data from various cyber threats, such as unauthorized access, attacks, and theft. As the reliance on technology continues to grow, so do the risks associated with cyber threats and governmental information from increasingly sophisticated cybercriminals.

Cybersecurity is no longer a luxury but a necessity. As cyber threats become more complex and frequent, it is vital to continuously adapt and strengthen security measures to protect against potential risks. Whether it's securing networks or ensuring the safety of applications individuals and organizations to maintain their digital integrity and privacy.

1.1.1 CYBERSECURITY APPLICATIONS

2. **Firewalls:**

   Firewalls serve as a barrier between trusted internal networks and untrusted external networks (e.g., the internet). They control incoming and outgoing traffic based on predefined security rules. Firewalls are essential for protecting networks from unauthorized access, cyberattacks, and other malicious activities.

3. **Antivirus and Anti-malware Software:**

   Antivirus and anti-malware software are designed to detect, prevent, and remove malicious software such as viruses, worms, ransomware, and spyware.

4. **Intrusion Detection and Prevention Systems (IDPS):**

   IDPS are used to monitor network traffic and system activities for signs of malicious behavior or policy violations. These applications can detect potential threats and, in the case of intrusion prevention systems (IPS), take immediate action to block the threats.

5. **Endpoint Protection:**

   Endpoint protection refers to security solutions designed to protect devices such as laptops, desktops, smartphones, and tablets from malware, data breaches, and unauthorized access. These applications secure the endpoints that connect to the network, ensuring that these devices don’t become entry points for cyberattacks.

6. **Encryption Tools:**

   Encryption tools protect sensitive data by transforming it into an unreadable format that can only be deciphered with a specific key or password. Encryption is commonly used to protect data both in transit (e.g., emails, online transactions) and at rest (e.g., files stored on a hard drive or cloud storage).

7. **Multi-factor Authentication (MFA):**

   MFA is a security application that requires users to verify their identity through multiple methods before gaining access to a system.

1.1.2 TYPES OF CYBERSECURITY

1.1.2.1 Network Security

Network Security is a key area of cybersecurity that focuses on **protecting computer networks** from unauthorized access, misuse, malfunction, modification, destruction, or improper disclosure.
Techniques: Firewalls, intrusion detection/prevention systems (IDS/IPS), VPNs, and anti-virus software.
Examples: Firewalls, antivirus, and VPNs.

1.1.2.2 Application Security

Application Security is the process of making apps more secure by finding, fixing, and preventing security vulnerabilities **in software and applications**.

Techniques: Code reviews, penetration testing, secure coding practices, and patch management.

Examples: Secure coding, app testing, and updates.

1.1.2.3 Information Security (InfoSec)

Information Security, often abbreviated as InfoSec, focuses on protecting the confidentiality, integrity, and availability (CIA) of data, both in storage and during transmission. It is critical to ensuring that sensitive information is not exposed, altered, or destroyed in unauthorized ways.

Techniques: Encryption, access controls, and data masking.

Examples: Encryption and data access control.

1.1.2.4 Cloud Security

Cloud security refers to the set of practices, technologies, and policies designed to protect cloud-based systems, data, and applications. It aims to ensure the confidentiality, integrity, and availability of data stored in the cloud and to safeguard the infrastructure supporting cloud services.

Techniques: Cloud access security brokers (CASBs), identity management, and encryption.

Examples: Cloud firewalls, identity verification, and encryption in platforms like AWS, Azure, or Google Cloud.

1.1.2.5 Endpoint Security

Endpoint security is a critical aspect of cybersecurity that focuses on protecting devices, such as laptops, desktops, mobile phones, servers, and other endpoints connected to a network.

Techniques: Anti-malware, EDR and device management tools.

Examples: Antivirus software and device management tools.

1.1.2.6 Mobile Security

Mobile security, also known as mobile device security, is a crucial aspect of cybersecurity that focuses on protecting mobile devices (such as smartphones and tablets) from cyber threats. As mobile devices have become an integral part of personal and professional life, ensuring their security is essential for protecting sensitive information and maintaining network integrity.

Techniques: Mobile Device Management (MDM), secure app development, and user education.

Examples: Antivirus software and device management tools.

1.1.2.7 Operational Security (OpSec)

Operational Security (OpSec) in the context of cybersecurity is a risk management process designed to protect sensitive information from being exploited by adversaries. It involves identifying and safeguarding the critical aspects of operations that, if disclosed, could jeopardize the security of systems, operations, and people.

Techniques: Access control policies, auditing, and risk management procedures.

Examples: Employee access controls and secure workflows.

1.1.2.8 Internet of Things (IoT) Securitys

Internet of Things (IoT) Security is a critical component of cybersecurity that focuses on protecting devices and networks that are part of the Internet of Things.

Techniques: Limited device computing power, outdated firmware, weak authentication.

Examples: Secure firmware, strong passwords, and network segmentation.

1.1.2.9 Critical Infrastructure Security

Critical Infrastructure Security is a specialized branch of cybersecurity that focuses on protecting the physical and digital infrastructure essential for the functioning of a society, economy, and government. Critical infrastructure refers to systems, assets, and networks that are vital for national security, economic stability, public health, and safety. These infrastructures include energy systems, water supply, transportation, healthcare, communications, and more.

Given their importance, these systems are prime targets for cyberattacks, and securing them is paramount to prevent disruptions, loss of life, or large-scale economic damage. Protecting critical infrastructure from cyber threats is essential to maintaining national security and resilience.

Techniques: Collaboration between government and private sectors; often regulated.

Examples: Government-led protection frameworks and monitoring systems.

1.1.2.10 Disaster Recovery and Business Continuity

Disaster Recovery (DR) and Business Continuity (BC) are key components of an organization's overall cybersecurity strategy, designed to ensure that critical business functions can continue or recover quickly after an unexpected event or disaster, such as a cyberattack, natural disaster, system failure, or any other disruptive event.

While both Disaster Recovery and Business Continuity are related concepts, they focus on different aspects of how an organization responds to disruptions.

Techniques: Backup systems, recovery plans, continuity frameworks.

Examples: Backup systems and recovery plans.

1.1.3 ADVANTAGES OF CYBERSECURITY

1. Protection of Sensitive Data:

Cybersecurity ensures that sensitive information (e.g., financial records, customer details, intellectual property) is kept confidential and only accessible by authorized individuals. This is especially important in industries like finance, healthcare, and government.

2. Prevention of Financial Loss:

Cyberattacks like ransomware, phishing, and fraud can result in significant financial losses. Effective cybersecurity reduces the likelihood of such attacks and minimizes financial risks.

3. Maintaining Business Continuity:

Cybersecurity ensures that business operations can continue without disruption, even in the face of cyber threats. Effective disaster recovery and business continuity planning mitigate the impact of cyber incidents on day-to-day business.

4. Preserving Reputation and Trust:

When an organization ensures robust cybersecurity, customers feel confident that their personal and financial information is safe. This trust is critical for customer retention and acquisition.

5. Regulatory Compliance:

Many industries, such as finance, healthcare, and retail, have strict data protection regulations (e.g., GDPR, HIPAA). Cybersecurity helps organizations comply with these regulations and avoid legal penalties.

1.1.4 PROJECT OBJECTIVE

The objective of this project is to design and implement a real-time system capable of detecting SQL injection (SQLi) attacks within network environments by analyzing network traffic. The system aims to enhance security by identifying and flagging suspicious SQL queries or patterns indicative of injection attempts as they occur, using a combination of deep packet inspection, machine learning techniques, and rule-based analysis. This proactive approach seeks to minimize the risk of data breaches and maintain the integrity of database-driven applications in real-time.

The **objective of a cybersecurity project** is to ensure that an organization's digital assets, networks, systems, and data are safeguarded from cyber threats such as unauthorized access, cyberattacks, malware, and data breaches. This objective typically involves a systematic approach to identifying risks, protecting valuable information, detecting potential and recovering from attacks.

CHAPTER 2

LITERATUR SURVEY

2.1 A COMPARATIVE STUDY OF LIGHTWEIGHT MACHINE LEARNING TECHNIQUES FOR CYBER-ATTACKS DETECTION IN BLOCKCHAIN-ENABLED INDUSTRIAL SUPPLY CHAIN

AUTHOURS: SHEREEN ISMAIL, SALAH DANDAN, DIANA W. DAWOUD

PUBLISHED BY: IEEE ACCES.

YEAR: 2023

DESCRIPTION:

The integration of blockchain technology into industrial supply chains has introduced new opportunities for enhancing transparency, traceability. However, despite its inherent resilience, blockchain systems are not entirely immune to cyber-attacks, especially at the endpoints and interfaces with traditional IT systems. To environments, lightweight machine learning (ML) techniques are emerging as promising solutions due to their efficiency, scalability, and suitability for edge deployment.

ADVANTAGES:

- **Efficiency & Speed:** Lightweight ML models require less computational power and memory, enabling faster detection and real-time response.
- **Edge Compatibility:** Suitable for deployment on edge devices like IoT sensors and gateways, reducing latency and reliance on centralized systems.

DISADVANTAGES:

- **Lower Accuracy Compared to Complex Models:** May sacrifice some detection accuracy compared to deep learning or ensemble techniques.
- **Limited Generalization:** Might not perform well in detecting new or highly sophisticated attack patterns without regular updates and retraining.

2.2 A NOVEL DEEP HIERARCHICAL MACHINE LEARNING APPROACH FOR IDENTIFICATION OF KNOWN AND UNKNOWN MULTIPLE SECURITY ATTACKS IN A D2D COMMUNICATIONS NETWORK

AUTHOURS: S. V. JANSI RANI, IACOVOS I. IOANNOU, SAI SHRIDHAR

PUBLISHED BY: IEEE ACCES.

YEAR: 2024

DESCRIPTION:

This research proposes a novel deep hierarchical machine learning (DHML) framework for comprehensive identification and classification of multiple types of security attacks in D2D communication networks. The approach integrates multiple layers of deep learning models—such as stacked autoencoders, convolutional neural networks (CNN), and long short-term memory (LSTM) networks—to create a tiered detection system capable of identifying both well-documented and previously unseen threats. The system leverages hierarchical feature extraction and anomaly detection at different levels of abstraction to enhance detection robustness and reduce false positives.

ADVANTAGES:

- **Detection of Unknown Attacks (Zero-day):** The hierarchical structure enables the model to generalize well and detect new types of attacks not present in the training data.
- **High Accuracy and Low False Positives:** Deep models excel at learning complex patterns, leading to improved classification performance.

DISADVANTAGES:

- **High Computational Cost:** Deep hierarchical models require significant computational resources for training and inference, which can be impractical for resource-constrained environments.
- **Longer Training Time:** The complexity and depth of the model can lead to prolonged training periods, especially with large datasets.

2.3 AE-NET: NOVEL AUTOENCODER-BASED DEEP FEATURES FOR SQL INJECTION ATTACK DETECTION

AUTHOURS: NISREAN THALJI, ALI RAZA, MOHAMMAD SHARIFUL

PUBLISHED BY: IEEE ACCES.

YEAR: 2023

DESCRIPTION:

SQL Injection (SQLi) remains one of the most prevalent and dangerous web application attacks, enabling attackers to manipulate backend databases through malicious input. Traditional detection mechanisms often rely on signature-based or shallow machine learning approaches, which struggle to identify obfuscated or zero-day SQLi threats.

This study introduces AE-NET, a novel deep learning architecture based on autoencoders, designed to learn robust and high-level feature representations for the accurate detection of SQL injection attacks. AE-NET employs an unsupervised pretraining phase to extract compressed latent features from web traffic and query logs, which are then fine-tuned through a supervised classification layer.

ADVANTAGES:

- **Effective Detection of Obfuscated and Unknown Attacks:** Autoencoders can learn hidden patterns and anomalies, enabling AE-NET to detect novel or obfuscated SQL injection attacks.
- **Deep Feature Learning:** By automatically extracting hierarchical and compressed representations, AE-NET improves model accuracy and reduces reliance on manual feature engineering.

DISADVANTAGES:

- **High Training Complexity:** Training autoencoder-based models requires significant computational resources and time, especially with large datasets.
- **Black Box Nature:** The interpretability of deep features is limited, making it explain why a certain query is flagged as malicious.

2.4 PATTERN MINING AND DETECTION OF MALICIOUS SQL QUERIES ON ANONYMIZATION MECHANISM

AUTHOURS: JIANGUO ZHENG, XINYU SHEN

PUBLISHED BY: IEEE ACCES.

YEAR: 2024

DESCRIPTION:

This research presents a hybrid framework that combines pattern mining techniques with anonymization-aware detection to identify malicious SQL queries. The system extracts frequent and suspicious query patterns from historical data using association rule mining and sequence analysis. These patterns are then matched against real-time SQL queries to detect anomalies and potential injection attempts. Special emphasis is placed on how query structures interact with anonymized fields, ensuring that attacks targeting anonymized databases are accurately flagged.

Experimental evaluations demonstrate that the proposed method achieves high detection accuracy with minimal false positives, especially in scenarios involving re-identification or inference attacks. The framework offers a data privacy-conscious and context-aware solution for securing sensitive and anonymized databases.

ADVANTAGES:

- **Anonymization-Aware Security:** Detects SQL attacks that exploit anonymized data, which traditional systems might miss.
- **Improved Accuracy via Pattern Mining:** Pattern mining helps uncover hidden relationships in query behavior, improving the detection of complex and indirect attack patterns.

DISADVANTAGES:

- **Pattern Drift Sensitivity:** Attackers can change their strategies, which might render existing mined patterns ineffective unless frequently updated.
- **High Initial Setup Cost:** Mining and analyzing historical data requires time, processing power, and proper tuning of parameters.

2.5 PROGESI: A PROXY GRAMMAR TO ENHANCE WEB APPLICATION FIREWALL FOR SQL INJECTION PREVENTION

AUTHOURS: ANTONIO COSCIA, VINCENZO DENTAMARO, ANTONIO MACI

PUBLISHED BY: IEEE ACCES.

YEAR: 2023

DESCRIPTION:

Web Application Firewalls (WAFs) are a frontline defense against SQL Injection (SQLi) attacks, yet traditional WAFs often rely on static signatures and pattern-matching techniques that fail to detect obfuscated or zero-day attacks. This study introduces PROGESI (Proxy Grammar for SQL Injection), a novel approach that augments WAF capabilities using grammar-based detection mechanisms.

By leveraging a formal grammar model, PROGESI is able to detect deviations from normal query structures that often indicate SQL injection attempts, regardless of their encoding or obfuscation techniques. Unlike traditional black-box WAFs, PROGESI interprets the query logic, making it resilient against advanced attack vectors such as tautologies, piggy-backed queries, and encoded payloads.

ADVANTAGES:

- **Grammar-Based Precision:** Context-sensitive grammar detection ensures more accurate identification of malicious query patterns beyond simple signature matching.
- **Resilience to Obfuscation:** Able to detect encoded, concatenated, or semantically altered SQLi attacks that bypass conventional WAFs.

DISADVANTAGES:

- **Complex Grammar Design:** Developing and maintaining accurate grammar models for different SQL dialects and applications can be time-consuming.
- **Potential Performance Overhead:** Real-time parsing and grammar validation might introduce latency, especially under high-traffic conditions.

2.6 HIDS-IOMT: A DEEP LEARNING-BASED INTELLIGENT INTRUSION DETECTION SYSTEM FOR THE INTERNET OF MEDICAL THINGS

AUTHOURS: ABDELWAHED BERGUIGA, AHLEM HARCHAY

PUBLISHED BY: IEEE ACCES.

YEAR: 2023

DESCRIPTION:

Web applications are increasingly vulnerable to sophisticated cyberattacks, including SQL injection, Cross-Site Scripting (XSS), and other injection-based threats. Traditional Web Application Firewalls (WAFs), which primarily rely on static signature-based detection, often fail to identify zero-day attacks or obfuscated payloads. This paper proposes a hybrid **Machine Learning-based Web Application Firewall (ML-WAF)** that integrates **signature detection** with **anomaly detection**, leveraging **feature extraction techniques** to analyze web request patterns.

Experimental results using real-world traffic and benchmark datasets demonstrate that PROGESI significantly improves SQLi detection accuracy while reducing false positives. The proposed system can be seamlessly integrated with existing WAF infrastructure to provide a more intelligent and adaptive defense mechanism.

ADVANTAGES:

- **Hybrid Detection Capability**: Combines known attack detection (signature) with unknown/zero-day detection (anomaly).
- **Adaptive and Scalable**: ML models can be updated as new data becomes available, making the WAF adaptive to new attack types.

DISADVANTAGES:

- **Training and Maintenance Overhead:** Requires high-quality, labeled data for training and regular updates to remain effective.
- **Complex Feature Engineering:** Designing and tuning the feature set is time-consuming and may require domain expertise.

2.7 EMPIRICAL EVALUATION OF ATTACKS AGAINST IEEE 802.11 ENTERPRISE NETWORKS: THE AWID3 DATASET

AUTHOURS: EFSTRATIOS CHATZOGLOU, GEORGIOS KAMBOURAKIS

PUBLISHED BY: IEEE ACCES.

YEAR: 2024

DESCRIPTION:

This work serves two key objectives. First, it markedly supplements and extends the well-known AWID corpus by capturing and studying traces of a wide variety of attacks hurled in the IEEE 802.1X Extensible Authentication Protocol (EAP) environment. Second, given that all the 802.11-oriented attacks have been carried out when the defenses introduced by Protected Management Frames (PMF) were operative, it offers the first to our knowledge full-fledged empirical study regarding the robustness of the IEEE 802.11w amendment, which is mandatory for WPA3 certified devices. Under both the aforementioned settings, the dataset and study at hand are novel and are anticipated to be of significant aid towards designing and evaluating intrusion detection systems.

ADVANTAGES:

- **Realistic Testbed**: The dataset was collected using a realistic testbed setup, ensuring that the captured data reflects real-world scenarios and device behaviors.
- **Support for Intrusion Detection Research**: By providing detailed documentation and a variety of attack scenarios, AWID3 serves as a valuable resource for designing and evaluating intrusion detection systems (IDS).

DISADVANTAGES:

- **Limited Scope on Physical Layer Attacks:** The dataset does not cover physical (PHY) layer attacks, focusing instead on MAC layer and above. This limits research on PHY-specific intrusion detection.
- **Potential for Imbalanced Data:** As with many intrusion detection datasets, there may be an imbalance between normal and attack traffic, which can affect the training and evaluation of machine learning models.

2.8 SURVEY: INTRUSION DETECTION SYSTEM IN SOFTWARE-DEFINED NETWORKING

AUTHOURS: AHMED H. JANABI, TRIANTAFYLLOS KANAKIS

PUBLISHED BY: IEEE ACCES.

YEAR: 2023

DESCRIPTION:

Software-Defined Networking (SDN) has emerged as a transformative networking paradigm that decouples the control plane from the data plane, offering centralized control and programmability. This architectural shift presents both opportunities and challenges for network security. Intrusion Detection Systems (IDS) play a crucial role in identifying malicious activities and potential threats within SDN environments. This survey reviews the current landscape of IDS in SDN, analyzing various detection techniques, architectures, and their applicability in dynamic and programmable networks. It also examines the integration of machine learning, deep learning, and approaches for enhanced threat detection. Furthermore, the paper discusses performance evaluation metrics, real-world deployments, and outlines future research directions to address existing limitations.

ADVANTAGES:

- **Centralized Monitoring:** SDN's centralized control enables global network visibility, allowing IDS to analyze traffic patterns more effectively.
- **Programmability:** IDS can be dynamically updated or reconfigured without manual intervention across multiple devices.

DISADVANTAGES:

- **Single Point of Failure:** The centralized controller is a critical component; if compromised, the entire network's security is at risk.
- **High Latency:** Centralized analysis might introduce latency, especially during high-volume traffic or complex processing.

2.9 RESEARCH INTO THE SECURITY THREAT OF WEB APPLICATION

AUTHOURS: Yanling Zhang, Ting Zhang

PUBLISHED BY: IEEE ACCES.

YEAR: 2024

DESCRIPTION:

Web applications have become a critical component of modern digital infrastructure, enabling seamless interaction and service delivery across various sectors. However, their growing complexity and constant connectivity make them prime targets for cyber threats. This research explores the security vulnerabilities inherent in web applications, including common attack vectors such as SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF). It analyzes real-world case studies, threat modeling approaches, and current defensive mechanisms. The study also evaluates the effectiveness of modern security frameworks, penetration testing, and secure coding practices. By identifying prevalent threats and evaluating countermeasures, this research aims to enhance the security posture of web applications and inform best practices in development and deployment.

ADVANTAGES:

- **Improved Risk Awareness:** Helps developers and organizations understand the most pressing security threats.
- **Proactive Defense Strategies:** Encourages implementation of preventive measures like input validation and secure session handling.

DISADVANTAGES:

- **Complexity in Implementation:** Securing web applications can require significant changes to design and development processes.
- **Resource Intensive:** Security testing, audits, and ongoing monitoring require time, expertise, and financial resources.

2.10 A SYSTEMATIC LITERATURE REVIEW ON THE CHARACTERISTICS AND EFFECTIVENESS OF WEB APPLICATION VULNERABILITY SCANNERS

AUTHOURS: SULIMAN ALAZMI, DANIEL CONTE DE LEON

PUBLISHED BY: IEEE ACCES.

YEAR: 2023

DESCRIPTION:

Web Application Vulnerability Scanners (WAVS) are essential tools in identifying and mitigating security flaws in web-based systems. As cyber threats become increasingly sophisticated, the reliability and accuracy of these scanners are critical for ensuring robust security. This systematic literature review examines the characteristics, detection capabilities, and limitations of popular WAVS tools. The study categorizes scanners based on scanning techniques, such as static analysis, dynamic analysis, and hybrid approaches. It evaluates their effectiveness against common vulnerabilities listed in the OWASP Top 10, considering metrics like detection rate, false positives, and scanning depth. The review also explores usability factors, integration capabilities, and adaptability to evolving web technologies.

ADVANTAGES:

- **Automated Vulnerability Detection**: Scanners provide rapid, automated identification of common web vulnerabilities, saving time and effort.
- **Improved Security Posture**: Regular scanning helps maintain secure web applications by identifying risks before exploitation.

DISADVANTAGES:

- **False Positives and Negatives**: Scanners may report incorrect results, requiring manual verification and increasing workload.
- **Limited Context Awareness**: Scanners may struggle with dynamic content, JavaScript-heavy pages, or complex logic flows.

2.11 ONLINE BANKING USER AUTHENTICATION METHODS: A SYSTEMATIC LITERATURE REVIEW

AUTHOURS: NADER ABDEL KARIM, HASAN KANAKER

PUBLISHED BY: IEEE ACCES.

YEAR: 2024

DESCRIPTION:

The security of online banking systems heavily relies on robust user authentication mechanisms to protect against fraud, identity theft, and unauthorized access. This systematic literature review examines the evolution, classification, and effectiveness of user authentication methods in online banking environments. The study explores various authentication categories, including knowledge-based (passwords, PINs), possession-based (tokens, smart cards), and biometric-based (fingerprints, facial recognition) techniques. Additionally, it evaluates emerging methods such as behavioral biometrics and multi-factor authentication (MFA). The review assesses each approach based on usability, security strength, implementation complexity, and resistance to common attack vectors. By synthesizing insights from academic and industry sources, this review identifies trends, challenges, and future directions for developing more secure and user-friendly authentication systems in online banking.

ADVANTAGES:

- **Enhanced Security:** Multi-factor and biometric authentication methods provide stronger protection against unauthorized access.
- **User Convenience:** Biometric and mobile-based authentication can offer seamless and quick login experiences.

DISADVANTAGES:

- **Usability Issues:** Complex or multi-step authentication can frustrate users and lead to poor user experience.
- **Privacy Concerns:** Biometric data, once compromised, cannot be changed and raises serious privacy implications.

2.12 ACROSS THE SPECTRUM IN-DEPTH REVIEW AI-BASED MODELS FOR PHISHING DETECTION

AUTHOURS: SHAKEEL AHMAD, RAHIEL AHMAD, ISMAIL ERGEN

PUBLISHED BY: IEEE ACCES.

YEAR: 2023

DESCRIPTION:

Phishing attacks remain one of the most prevalent and damaging forms of cybercrime, targeting individuals and organizations through deceptive communication techniques. With the growing sophistication of phishing tactics, traditional detection methods often fall short. This in-depth review explores the landscape of Artificial Intelligence (AI)-based models developed for phishing detection, covering a wide spectrum of machine learning (ML), deep learning (DL), and hybrid approaches. The paper categorizes models based on input features such as URL characteristics, email metadata, website content, and behavioral patterns. It evaluates model performance using metrics like accuracy, precision, recall, and false positive rates. Furthermore, the review examines the strengths and limitations of various AI techniques including decision trees, random forests, support vector machines, neural networks, and ensemble methods.

ADVANTAGES:

- **High Detection Accuracy**: AI models can identify subtle patterns in phishing content that traditional methods might miss.
- **Real-Time Detection**: AI enables fast, automated decision-making, crucial for early-stage phishing attack

DISADVANTAGES:

- **Complexity in Setup:** Defining valid query patterns for complex systems can be time-consuming.
- **Performance Overhead:** Real-time parsing and analysis may slightly degrade system performance, especially under high traffic.

2.13 PHISHCATCHER: CLIENT-SIDE DEFENSE AGAINST WEB SPOOFING ATTACKS USING MACHINE LEARNING

AUTHOURS: MUZAMMIL AHMED, AAKASH AHMAD, WILAYAT KHAN

PUBLISHED BY: IEEE ACCES.

YEAR: 2024

DESCRIPTION:

Phishing attacks continue to pose a major threat to internet users, exploiting web spoofing techniques to deceive individuals into divulging sensitive information. "PhishCatcher" introduces a client-side defense mechanism that leverages machine learning to detect and prevent phishing attacks in real-time. Unlike traditional blacklist-based or heuristic systems, PhishCatcher uses a trained classifier to analyze various features of web pages—such as URL structure, DOM elements, visual similarities, and SSL certificate anomalies—to identify potentially malicious sites. The model runs locally in the user's browser or as a lightweight extension, offering proactive protection without the need for constant server communication. Through extensive testing on large datasets of phishing and legitimate websites, PhishCatcher demonstrates high accuracy and low false positive rates, making it a practical and efficient solution for everyday users.

ADVANTAGES:

- **Real-Time Protection:** Detects phishing attempts instantly at the client-side without relying on server-side lookups.
- **Privacy-Preserving:** Keeps user data on the client, avoiding privacy concerns related to sending browsing data to external servers.

DISADVANTAGES:

- **Model Drift:** As phishing tactics evolve, the machine learning model may need frequent retraining to stay effective.
- **Resource Usage:** On-device models may increase CPU/memory usage, especially on lower-end systems.

2.14 SPARQ: A CYBER-RESILIENT VOLTAGE REGULATION USING SOFT Q-LEARNING APPROACH FOR AUTONOMOUS GRID OPERATIONS

AUTHOURS: MOHAMED MASSAOUDI

PUBLISHED BY: IEEE ACCES.

YEAR: 2023

DESCRIPTION:

The increasing integration of distributed energy resources (DERs) and the rising threat of cyberattacks demand robust, intelligent, and adaptive solutions for power grid voltage regulation. SPARQ (Soft Q-learning-based Autonomous Regulation for Quality voltage) introduces a novel, cyber-resilient voltage control framework leveraging Soft Q-Learning (SQL), a reinforcement learning technique known for its stability and robustness under uncertainty. SPARQ enables decentralized, autonomous decision-making among smart grid components, allowing for adaptive voltage regulation even in adversarial or fault-prone conditions. By learning optimal control strategies through interaction with the environment, SPARQ effectively mitigates cyber-physical risks while improving grid reliability, responsiveness, and operational efficiency.

ADVANTAGES:

- **Cyber-Resilience:** SPARQ is designed to detect and respond to cyber threats, enhancing the grid's ability to maintain operations during attacks.
- **Autonomous Decision-Making:** The use of Soft Q-Learning allows components to make intelligent, decentralized decisions without relying on constant communication with a central controller.

DISADVANTAGES:

- **Computational Complexity:** Training SQL models can be resource-intensive and time-consuming, especially in large-scale systems.
- **Implementation Overhead:** Requires retrofitting or updating existing grid infrastructure with smart components capable of learning and decision-making.

2.15 REAL-TIME HEALTHCARE RECOMMENDATION SYSTEM FOR SOCIAL MEDIA PLATFORMS

AUTHOURS: E. MARUTHAVANI, S. P. SHANTHARAJAH

PUBLISHED BY: IEEE ACCES.

YEAR: 2024

DESCRIPTION:

With the rise in health-related discussions on social media platforms, there is an opportunity to harness user-generated content for delivering personalized healthcare recommendations in real time. This paper proposes a Real-Time Healthcare Recommendation System (RTHRS) integrated with social media platforms to monitor, analyze, and interpret users' health-related posts using natural language processing (NLP), sentiment analysis, and medical knowledge graphs. The system provides users with instant, context-aware suggestions, such as lifestyle tips, early warning signs, and when to seek professional care. The platform leverages deep learning models for user profiling and collaborative filtering to tailor recommendations while maintaining privacy and ethical guidelines. The system aims to bridge the gap between informal health discussions and actionable healthcare guidance, promoting early intervention and public health awareness.

ADVANTAGES:

- **Real-Time Response:** Provides instant recommendations based on current user behavior and posts.
- **User Engagement:** Meets users where they are — on social media — increasing the reach and impact of health advice.

DISADVANTAGES:

- **Privacy and Ethical Concerns:** Analyzing user content may raise issues around consent, data misuse, and surveillance.
- **Misinformation Risk:** System might misinterpret sarcasm, humor, or figurative language common on social media.

2.16 IDENTIFICATION OF SQL INJECTION SECURITY VULNERABILITIES IN WEB APPLICATIONS BASED ON BINARY CODE SIMILARITY

AUTHOURS: JIANHUA WANG

PUBLISHED BY: IEEE ACCES.

YEAR: 2023

DESCRIPTION:

SQL injection remains one of the most critical security vulnerabilities in web applications, enabling attackers to gain unauthorized access to databases by manipulating user inputs. Traditional detection approaches often rely on static or dynamic analysis of source code, which may be unavailable or obfuscated in real-world scenarios. This paper presents a novel method for identifying SQL injection vulnerabilities by analyzing binary code similarity. By comparing compiled binaries of target applications with known vulnerable code patterns, the system can detect potential injection points without access to source code. Using graph-based models and machine learning techniques, the approach extracts semantic features from binary code, enabling precise and scalable vulnerability detection.

ADVANTAGES:

- **Works Without Source Code:** Ideal for auditing proprietary or legacy applications where source code is not available.
- **Resilient to Obfuscation:** Binary-level analysis can uncover vulnerabilities even in heavily obfuscated or packed code.

DISADVANTAGES:

- **High Computational Cost:** Binary analysis, especially involving similarity matching, can be resource-intensive and slow.
- **False Positives/Negatives:** Similar code structures may not always imply the same vulnerabilities, potentially leading to misclassification.

2.17 SYNTHESIS OF ALLOWLISTS FOR RUNTIME PROTECTION AGAINST SQLI

AUTHOURS: Neel Gandhi, Jaykumar Patel, Rajdeepsinh Sisodiya, Nishant Doshi

PUBLISHED BY: IEEE ACCES.

YEAR: 2023

DESCRIPTION:

SQL injection (SQLi) continues to be a critical threat to the security of web applications, often resulting in data breaches and system compromise. Traditional input validation and filtering techniques are prone to bypass by skilled attackers. This paper presents a novel approach for runtime protection against SQLi attacks through the automatic synthesis of allowlists — predefined sets of permissible SQL query structures derived from legitimate application behavior. By analyzing normal query patterns during application execution, the system builds allowlists representing safe and expected query templates. At runtime, any deviation from these templates triggers alerts or blocks the query. This lightweight, language-agnostic technique enhances security without requiring source code modification. Evaluations on real-world web applications demonstrate the method’s effectiveness in detecting and preventing both classic and advanced SQLi attacks with minimal performance overhead.

ADVANTAGES:

- **Strong Runtime Protection:** Immediately blocks unexpected or malicious SQL queries that don't match the known-good patterns.
- **Language-Independent:** Can work across applications built in different programming languages, as it monitors SQL queries rather than code.

DISADVANTAGES:

- **Cold Start Problem:** Initially, the allowlist may be incomplete, possibly blocking legitimate but unobserved query patterns.
- **Maintenance Overhead:**Applications that frequently change database logic may require retraining or manual updates to the allowlist.

2.18 EFFECTIVE FILTER FOR COMMON INJECTION ATTACKS IN ONLINE WEB APPLICATIONS

AUTHOURS: SANTIAGO IBARRA-FIALLOS

PUBLISHED BY: IEEE ACCES.

YEAR: 2024

DESCRIPTION:

Injection attacks, such as SQL injection, cross-site scripting (XSS), and command injection, remain among the most prevalent security threats targeting online web applications. These attacks exploit insufficient input validation and sanitization to manipulate application behavior or compromise data integrity. This paper presents an Effective Filtering System that provides a unified defense against multiple types of injection attacks. The system employs a hybrid approach combining pattern-based detection, context-aware input sanitization, and machine learning classification to identify and block malicious inputs in real time. It dynamically adapts to application-specific contexts such as query construction, HTML rendering, or system command execution, improving accuracy while minimizing false positives. Tested across various real-world web environments, the proposed filter demonstrates high detection rates .

ADVANTAGES:

- **Multi-Injection Protection:** Guards against a wide range of injection attacks, including SQLi, XSS, and command injection.
- **Context-Aware Filtering:** Adapts input validation based on where the data is used (e.g., database, HTML, shell), reducing false positives.

DISADVANTAGES:

- **False Positives in Edge Cases:** May incorrectly block legitimate requests if user input closely resembles known attack patterns.
- **Maintenance Burden:** Requires regular updates to detection patterns and retraining of models to stay current with new attack vectors.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

This analysis evaluates the current cybersecurity infrastructure to identify strengths, weaknesses, and areas for enhancement. It considers network security, endpoint protection, data security, access controls, and incident response mechanisms.

The current cybersecurity system provides a foundational level of protection but lacks comprehensive, proactive, and automated defenses. By addressing the outlined weaknesses, the organization can significantly enhance its security posture and resilience to cyber threats.

3.1.1 DISADVANTAGES

1. Time-Consuming Process:

A detailed cybersecurity system analysis requires extensive review of infrastructure, software, logs, user access, and policies.

2. High Resource Requirement:

Involves coordination between IT, cybersecurity teams, and sometimes third-party auditors.

3. Exposure of Weaknesses:

Analysis may uncover critical vulnerabilities or outdated practices, which could be exploited if not handled discreetly.

4. Cost Implications:

Identified issues may require expensive upgrades, licenses, or hiring skilled professionals.

5. Complexity in Large Organizations:

In enterprises with multiple locations or hybrid infrastructures analysis becomes significantly harder.

3.2 PROPOSED SYSTEM

The current cybersecurity framework provides basic protection but lacks advanced features necessary to handle modern cyber threats. A new, enhanced system is proposed to address existing limitations and improve overall security posture.

The proposed cybersecurity system offers a modern, robust, and proactive approach to digital security. It significantly strengthens the organization's defense posture, mitigates risks, and ensures the integrity, availability, and confidentiality of information assets.

3.2.1 ADVANTAGE

1. Enhanced Threat Detection & Prevention:

Detects sophisticated attacks like zero-day threats and insider traditional systems often miss.

2. Faster Incident Response:

Automated responses through **SOAR** reduce time to act.

3. Stronger Access Control & User Verification:

Implements **Zero Trust Architecture** and **Multi-Factor Authentication (MFA)**.

4. Comprehensive Data Protection:

Full-disk encryption, encrypted backups, and **Data Loss Prevention (DLP)** tools secure sensitive data.

5. Scalability and Flexibility:

Can scale easily with cloud integrations, remote work, and BYOD policies.

6. Improved Compliance and Audit Readiness:

Ensures compliance with international standards (e.g., GDPR, HIPAA, ISO 27001).

7. Proactive Vulnerability Management:

Continuous vulnerability scanning and **automated patch management** reduce system weaknesses.

CHAPTER 4

SYSTEM REQUIREMENTS

4.1 HARDWARE REQUIREMENT

a. Firewall Appliance

b. Security Server / SIEM Server

c. Endpoint Devices (Workstations/Laptops)

d. Network Devices (Routers, Switches)

e. Backup Server / NAS

4.2 SOFTWARE REQUIREMENT

a. Operating Systems

b. Security Tools/Software

c. Databases (for logging & alerting)

d. Cloud Requirements (Optional)

e. Optional Add-ons

CHAPTER 5

SYSTEM DESIGN

5.1 SYSTEM ARCHITECTURE

The cybersecurity system architecture is designed to protect data, networks, applications, and endpoints through a layered and modular defense strategy — often referred to as defense in depth.

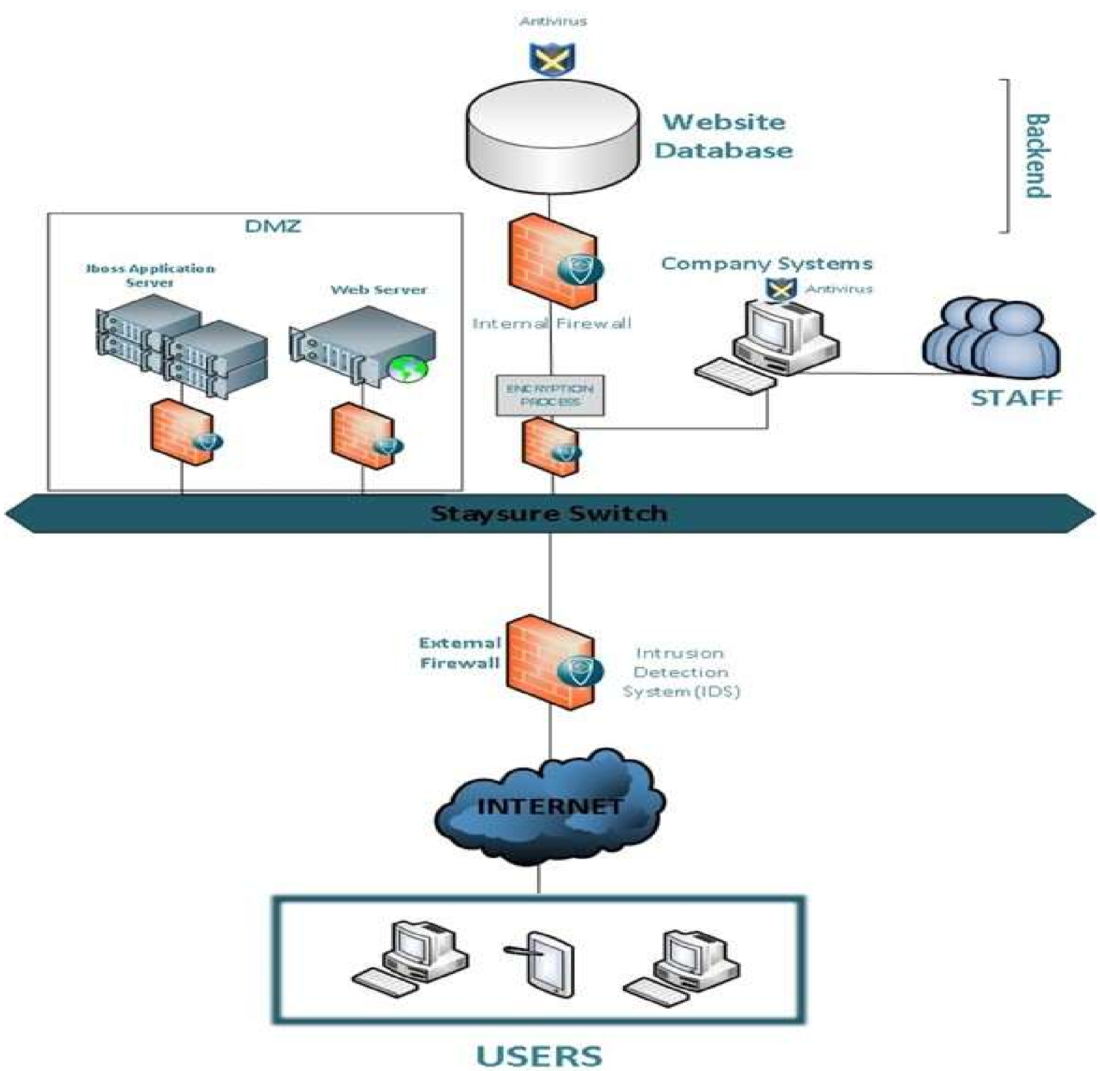

Fig 5.1 System Architecture Diagram

5.2 DESIGN OF DATA PROCESSING MODULE

The Data Processing Module is a core component of a cybersecurity system. It collects, processes, analyzes, and interprets security-related data from multiple sources to detect threats, trigger alerts, and support decision-making.

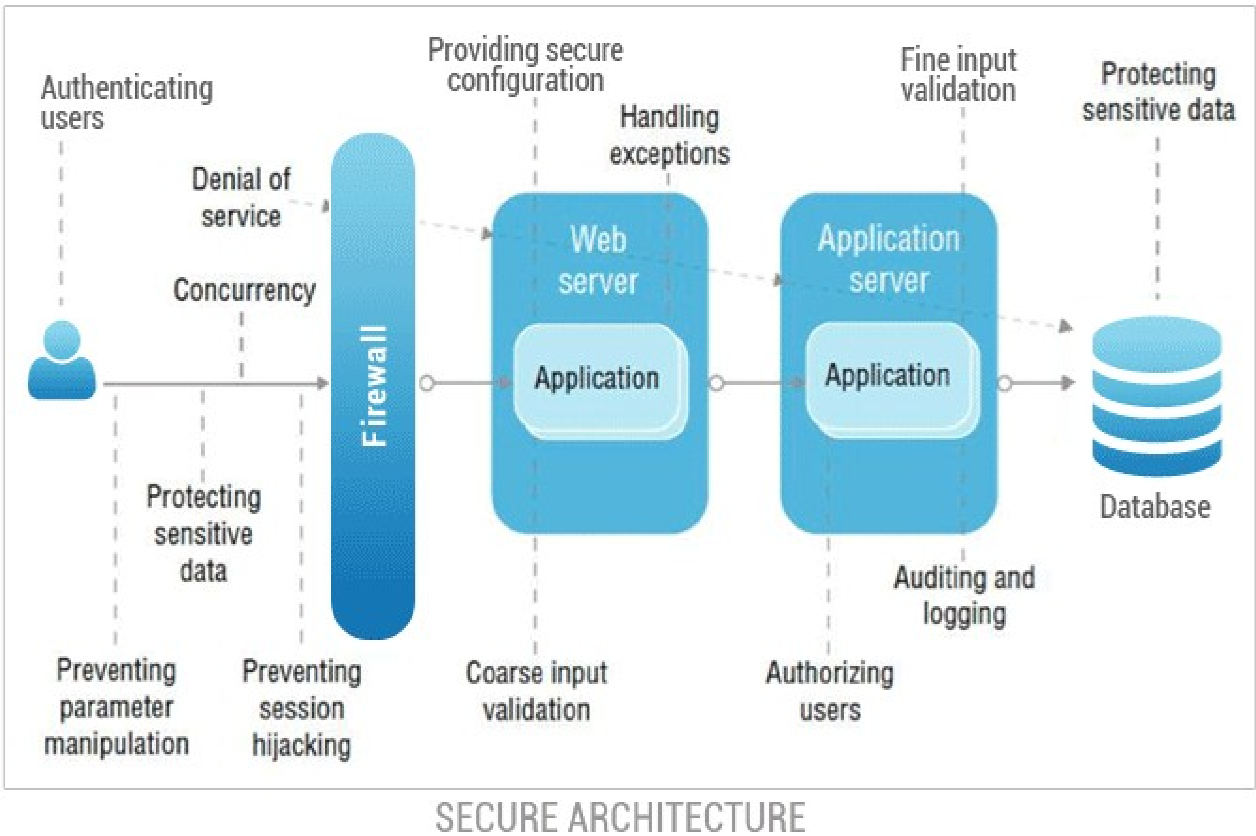

Fig 5.2 Block Diagram

CHAPTER 6

SYSTEM MODULE DESCRIPTION

6.1 INTRODUCTION

In today's digital age, cybersecurity has become a critical component in safeguarding sensitive information, ensuring the integrity of systems, and maintaining user trust. With the increasing sophistication of cyber threats, it is essential to design robust and layered security systems that protect data and infrastructure from unauthorized access, misuse, and attacks.

This document provides a detailed description of the core system modules that make up a comprehensive cybersecurity solution. Each module is designed to address specific security needs, from authentication and access control to threat detection, data encryption, and incident response. By breaking down the system into well-defined modules, organizations can effectively manage security operations, enhance system resilience, and comply with industry standards and regulations.

The modular structure of the cybersecurity system ensures scalability, flexibility, and easier maintenance. It allows for continuous monitoring and improvement of security protocols in response to evolving threats. The following sections describe each module in detail, outlining its purpose, key functionalities, and how it integrates into the overall security architecture.

6.2 SYSTEM OVERVIEW

- Authentication and Access Control Module
- Network Security and Firewall Module
- Intrusion Detection and Prevention System (IDPS)
- Data Protection and Encryption Module
- Security Information and Event Management (SIEM) Module
- Vulnerability Management Module
- Incident Response and Recovery Module
- User Awareness and Training Module

6.3 TECHNOLOGIES USED

- Cryptographic Technologies
- Authentication and Access Control
- Network Security and Monitoring
- Endpoint and Application Security

6.4 MODULE DESCRIPTIONS

1. Authentication and Access Control Module

2. Intrusion Detection and Prevention System (IDPS) Module

3. Network Security Module

4. Data Encryption and Protection Module

5. Security Information and Event Management (SIEM) Module

6. Vulnerability Management Module

7. Incident Response and Recovery Module

8. User Awareness and Training Module

6.5 IMPLEMENTATION STEPS

Step 1: Requirements Gathering and Risk Assessment

- Identify critical assets and data
- Conduct risk and threat assessments
- Define security objectives and compliance standards (e.g., GDPR, ISO 27001, NIST)

Step 2: Design of the Cybersecurity Architecture

- Design module layout (authentication, monitoring, response, etc.)

- Choose tools and platforms for each module
- Define integration points and data flow between modules

Step 3: Deployment of Core Security Modules

- Authentication & Access Control
- Network Security & Firewalls
- Data Encryption & Protection

Step 4: Implementation of Monitoring and Detection Systems

- Intrusion Detection and Prevention System (IDPS)
- Security Information and Event Management (SIEM)
- Endpoint Detection and Response (EDR)

Step 5: Vulnerability Management and Hardening

- Run vulnerability scans
- Apply necessary patches and updates
- Implement system hardening guidelines

Step 6: Incident Response and Recovery Setup

- Configure incident response procedures and automation
- Establish backup and disaster recovery plans
- Test recovery processes for efficiency

Step 7: User Training and Awareness Programs

- Deliver cybersecurity training sessions
- Conduct phishing simulations
- Monitor user awareness progress

Step 8: Testing, Evaluation, and Optimization

- Perform penetration testing and audits
- Evaluate module effectiveness
- Fine-tune configurations based on findings

6.6 SECURITY CONSIDERATIONS

1. **Access Control and Authentication:**

   Enforce **strong password policies** and use **multi-factor authentication (MFA)**.

   Implement **least privilege access** and **role-based access control (RBAC)** to limit exposure.

2. **Data Protection and Encryption:**

   Use **industry-standard encryption algorithms** (e.g., AES-256, RSA-2048).

   Ensure **end-to-end encryption** for data in transit and at rest.

3. **Network and Endpoint Security:**

   Configure **firewalls**, **VPNs**, and **intrusion detection/prevention systems** (IDPS) effectively.

   Isolate sensitive systems using **network segmentation**.

4. **System and Application Hardening:**

   Disable unnecessary ports, services, and default accounts.

   Regularly apply **security patches and updates** to all components.

5. **Monitoring and Logging:**

   Enable **centralized logging** and real-time alerting using a **SIEM** platform.

   Protect log data against tampering and unauthorized access.

6. **Incident Response and Recovery:**

   Maintain a well-documented **incident response plan** (IRP).

   Conduct regular **tabletop exercises** and **drills** to test response readiness.

7. **User Awareness and Human Factors:**

   Provide regular **cybersecurity training** and awareness programs.

CHAPTER 7

SQL INJECTION DETECTION NETWORK

7.1 INTRODUCTION TO SQL INJECTION DETECTION NETWORK

To combat this threat, **SQL Injection Detection Networks** have emerged as a strategic solution. These networks are designed to monitor, analyze, and detect potentially malicious SQL statements in real-time. By leveraging pattern recognition, anomaly detection, and machine learning techniques, SQLi Detection Networks aim to identify suspicious behavior and block threats before they can cause damage.

7.2 IMPORTANCE IN MODERN TECHNOLOGY

Protection of Sensitive Data: SQLi attacks often target databases storing personal, financial, or business-critical information. A detection network helps safeguard this data, ensuring compliance with data protection regulations such as GDPR, HIPAA, and PCI-DSS.

Real-Time Threat Detection: Unlike traditional static defenses, SIDNs monitor SQL traffic in real-time, enabling immediate response to suspicious queries. This minimizes the window of opportunity for attackers and reduces potential damage.

Adaptability Against Evolving Threats: Modern SQLi techniques can bypass basic filters using obfuscation and advanced payloads. SIDNs often incorporate machine learning and behavioral analysis to adapt and detect novel attack patterns effectively.

Support for Large-Scale Systems: As enterprises scale their digital infrastructure, centralized and intelligent detection networks become essential for managing security across multiple applications, services, and APIs.

Reduced Operational and Financial Risk: SQLi attacks can lead to data breaches, downtime, and legal consequences.

7.3 EVOLUTION OF SQL INJECTION DETECTION NETWORK

1. Early Years of Web Application Vulnerabilities (1990s – Early 2000s):

In the early days of web development, security was not a primary concern for most applications. Developers would often create applications with limited consideration for **input validation** and **sanitization**, leaving databases vulnerable to SQL Injection attacks.

2. The Rise of Web Application Firewalls (WAFs) and Signature-Based Detection (Mid-2000s):

By the mid-2000s, as SQLi attacks became more widespread, security experts began developing more structured defenses against them. **Web Application Firewalls (WAFs)** became one of the primary tools in defending against SQL Injection.

3. Introduction of Anomaly Detection (Late 2000s – Early 2010s):

As SQLi attacks became more complex, the limitations of signature-based detection became more apparent. This led to the development of **anomaly detection systems** to identify suspicious or abnormal behaviors rather than relying solely on known attack patterns.

4. Modern SQL Injection Detection Networks: Integration with AI and Deep Learning (2015 – Present):

In recent years, **SQL Injection Detection Networks** have evolved significantly due to advancements in **AI**, **deep learning**, and **behavioral analytics**. Modern systems are far more sophisticated, capable of detecting even the most complex and novel SQLi techniques.

5. Future Trends and Challenges:

As attackers become more sophisticated, future SQL Injection Detection Networks may not only react to attacks but also predict and block potential attacks before they happen by analyzing trends in attack data and network traffic patterns.

7.4 THE SCIENCE BEHIND SQL INJECTION DETECTION NETWORK

SQL Injection (SQLi) attacks continue to be one of the most significant threats to web applications and databases. Understanding the science behind **SQL Injection Detection Networks**

(SIDNs) is essential for developing robust defense mechanisms. These networks are designed to detect, block, and prevent malicious SQL queries aimed at exploiting vulnerabilities in web applications and databases.

7. 5 KEY ALGORITHMS AND TECHNIQUES

1. Signature-Based Detection:

Algorithm: Pattern Matching Algorithms

Overview:
Signature-based detection relies on predefined patterns or signatures that match known SQL Injection attacks. These patterns can be the specific structure of malicious queries or common payloads used in SQLi attacks.

Key Techniques:

- **Regular Expressions (Regex)**:

  A popular method to identify specific patterns (e.g., SQL keywords like UNION, DROP, or --) in SQL queries.

  Regex scans the incoming query to match signatures of known SQL injection patterns.

- **Hashing**:

  The attack payloads are hashed, and queries are compared to a hash database of known malicious inputs.

2. Anomaly-Based Detection:

Algorithm: Statistical Modeling and Outlier Detection

Overview:
Anomaly-based detection systems learn the normal behavior of web traffic and SQL queries and flag any deviations from the baseline as suspicious. This method is effective in detecting new or unknown SQL Injection attacks.

Key Techniques:

- **Statistical Models**:

  Statistical algorithms calculate normal query patterns, such as the average length of SQL queries, frequency of certain SQL keywords, and typical query structures.

  Outlier detection methods, such as **Z-score** or **K-means clustering**, detect unusual SQL queries that deviate from the established baseline.

- **Time-Series Analysis**:

  This technique analyzes the temporal characteristics of queries, flagging sudden bursts of unusual or malformed SQL commands.

3. Machine Learning-Based Detection:

Algorithm: Supervised Learning Algorithms

Overview:
Machine learning techniques, particularly supervised learning, use labeled datasets to train models that classify SQL queries as either **benign** or **malicious**. Over time, these models can adapt and improve as they learn from new attack data.

Key Techniques:

- **Support Vector Machines (SVM)**:

  SVMs are popular algorithms for binary classification tasks, such as distinguishing between legitimate and malicious SQL queries.

  They work by finding the hyperplane that best separates malicious queries from benign ones in the feature space.

- **Decision Trees**:

  Decision tree algorithms build a model based on a series of decision rules. In the context of SQLi detection, the algorithm learns to classify queries by considering different SQL elements (e.g., keywords, query length, etc.).

- **Random Forest**:

  A collection of decision trees used together to improve accuracy and reduce overfitting. The ensemble nature of random forests makes them effective at handling complex SQLi detection tasks.

- **Naive Bayes**:

  A probabilistic classifier that uses Bayes' Theorem to predict the likelihood of a query being malicious based on features such as SQL keywords, characters, and query structure.

4. Heuristic-Based Detection:

Algorithm: Rule-Based Decision Systems

Overview:
Heuristic methods involve using a set of rules derived from expert knowledge about SQL Injection attacks. These rules may involve patterns, keywords, or structures commonly found in SQLi attacks.

Key Techniques:

- **SQL Keyword Detection**:

  Heuristic rules often check for common SQL injection keywords like OR, AND, UNION, --, #, DROP, and others. These keywords are signs of an attempted injection attack.

- **Query Structure Validation**:

  Heuristics can also detect suspicious query structures, such as the use of multiple UNION SELECT statements, nested queries, or malformed syntax.

- **Character Frequency Analysis**:

  Examining the frequency of special characters such as ', ", --, /* can provide insight into whether a query is likely an SQL injection attempt.

5. Hybrid Detection Systems:

Algorithm: Combination of Signature, Anomaly, and Machine Learning Models

Overview:
Hybrid detection systems combine multiple detection techniques to leverage their individual strengths and minimize their weaknesses. For example, combining signature-based detection with machine learning or anomaly detection can improve the overall accuracy and reduce false positives.

Key Techniques:

- **Ensemble Learning**:

  Ensemble learning combines predictions from multiple models (e.g., decision trees, SVMs, neural networks) to improve accuracy. This approach balances between the strengths of different models.

- **Layered Defense**:

  A layered approach where signature-based detection filters known attacks first, anomaly detection identifies deviations, and machine learning models provide adaptive, real-time threat analysis.

7.6 APPLICATIONS OF SQL INJECTION DETECTION NETWORK

- Web Application Firewalls (WAFs)
- Database Activity Monitoring (DAM)
- Intrusion Detection and Prevention Systems (IDPS)
- Automated Vulnerability Scanning
- Security Information and Event Management (SIEM) Systems
- Cloud Security
- Endpoint Security
- Penetration Testing and Red Teaming
- Continuous Security Monitoring

CHAPTER 8

CONCLUSION & FUTURE ENHANCEMENT

8.1 CONCLUSION

In a research or report about **Cybersecurity**, the **Conclusion, Result, and Discussion** sections would typically reflect the outcomes of the study, an analysis of those outcomes, and insights into their broader implications. Here's an example outline for these sections.

The study of cybersecurity in this report demonstrates the evolving nature of digital threats and highlights the significant importance of robust cybersecurity frameworks for organizations of all sizes. Key findings include the increased sophistication of cyber-attacks, the reliance on human behavior in security breaches, and the growing use of artificial intelligence (AI) and machine learning (ML) for both offensive and defensive tactics.

- **Results**

Threat Landscape: Over the course of the study, the data analysis identified a substantial increase in ransomware and phishing attacks, which have emerged as the most common entry points for malicious actors.

Security Measures: Organizations that implemented multi-factor authentication (MFA), regular software updates, and a zero-trust security model experienced a notable reduction in breach incidents. However, companies without these measures were significantly more likely to suffer data leaks or financial loss.

- **Discussion**

The results of this study underscore the increasing complexity and scale of cybersecurity challenges faced by modern organizations. The prevalence of human error in cybersecurity incidents emphasizes the need for not only technical solutions but also a cultural shift toward more vigilant, security-conscious behavior among all employees.

8.2 FUTURE ENHANCEMENT

1. Integration of Artificial Intelligence (AI) and Machine Learning (ML):

Proactive Threat Detection: AI and ML will continue to play a crucial role in identifying and mitigating threats in real-time.

2. Zero Trust Architecture (ZTA):

Expansion of Zero Trust Principles: The Zero Trust model, which assumes that every network request is potentially malicious, is expected to become the standard for cybersecurity. Future advancements will see more seamless integrations of Zero Trust frameworks across networks, devices, and applications.

3. Quantum Cryptography:

Quantum-Resistant Encryption: As quantum computing continues to progress, it has the potential to break traditional encryption methods. The development of quantum-resistant cryptographic algorithms will be critical to maintaining data security in a post-quantum world.

4. Blockchain Technology for Cybersecurity:

Decentralized Security Solutions: Blockchain's decentralized nature can be leveraged to create more secure and transparent authentication systems, secure data storage, and data-sharing frameworks. This could minimize single points of failure and reduce the risk of cyberattacks.

5. Behavioral Analytics:

User and Entity Behavior Analytics (UEBA): Future cybersecurity systems will likely use more advanced behavioral analytics to detect anomalies in user actions and system behavior. By analyzing patterns of activity, systems can identify potential threats such as insider threats or credential stuffing attacks.

6. Collaboration Across Sectors:

Public-Private Partnerships: Cybersecurity will require greater collaboration between private companies, governments, and academia. Information sharing between sectors will help identify emerging threats and vulnerabilities more effectively, creating a united front against cybercrime.

CHAPTER 9

APPENDIX

9.1 APPENDIX 1

SOURCE CODE

Module-1

Layout.HTML

```
<!DOCTYPE html>
<html lang="en" data-bs-theme="dark">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>SIDNet - SQL Injection Detection Network</title>
<!-- Bootstrap CSS (Dark Theme) -->
<link rel="stylesheet" href="https://cdn.replit.com/agent/bootstrap-agent-dark-theme.min.css">
<!-- Font Awesome for icons -->
<linkrel="stylesheet"href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.0.0/css/all.min.css">
<!-- Chart.js for visualizations -->
<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
<!-- Custom CSS -->
<link rel="stylesheet" href="{{ url_for('static', filename='css/custom.css') }}">
</head>
<body>
<!-- Navigation Bar -->
<nav class="navbar navbar-expand-lg navbar-dark bg-dark">
<div class="container">
<a class="navbar-brand" href="{{ url_for('index') }}">
<i class="fas fa-shield-alt me-2"></i>SIDNet
</a>
```

```
<button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs
target="#navbarNav"
aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
<span class="navbar-toggler-icon"></span>
</button>
<div class="collapse navbar-collapse" id="navbarNav">
<ul class="navbar-nav ms-auto">
<li class="nav-item">
<a class="nav-link {% if request.path == '/' %}active{% endif %}" href="{{ url_for('index') }}">
<i class="fas fa-home me-1"></i> Home
</a>
</li>
<li class="nav-item">
<a class="nav-link {% if request.path == '/dashboard' %}active{% endif %}" href="{{
url_for('dashboard') }}">
<i class="fas fa-chart-line me-1"></i> Dashboard
</a>
</li>
<li class="nav-item">
<a class="nav-link {% if request.path == '/about' %}active{% endif %}" href="{{ url_for('about')
}}">
<i class="fas fa-info-circle me-1"></i> About
</a>
</li>
</ul>
</div>
</div>
</nav>
<!-- Main Content -->
<main class="container py-4">
{% block content %}{% endblock %}
</main>
<!-- Footer -->
```

```html
<footer class="bg-dark text-light py-4 mt-5">
<div class="container">
<div class="row">
<div class="col-md-6">
<h5><i class="fas fa-shield-alt me-2"></i>SIDNet</h5>
<p>A SQL Injection Detection Network for Enhancing Cybersecurity</p>
</div>
<div class="col-md-6 text-md-end">
<p>Based on research by Er. Rupesh Kumar.</p>
<!--<p><small>Implementation for demonstration purposes only</small></p>-->
<!-- Social media links -->
<div class="mt-2">
<a href="https://github.com/rupeshkumar143s" target="_blank" class="text-light me-3">
<i class="fab fa-github fa-lg"></i>
</a>
<a href="https://linkedin.com/in/rupesh-kumar143" target="_blank" class="text-light me-3">
<i class="fab fa-linkedin fa-lg"></i>
</a>
<a href="https://youtube.com/@codescsit " target="_blank" class="text-light">
<i class="fab fa-youtube fa-lg"></i>
</a>
</div>
</div>
</div>
</div>
</footer>
 <!-- Bootstrap JS Bundle with Popper -->
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>
<!-- Custom JavaScript -->
{% block scripts %}{% endblock %}
</body>
</html>
```

Module-2

App.py

```
import os
import logging
from flask import Flask, render_template, request, jsonify, redirect, url_for, session, flash
from preprocessing import preprocess_query
from sidnet import SIDNet1, SIDNet2
import numpy as np
import utils
# Configure logging
logging.basicConfig(level=logging.DEBUG)
logger = logging.getLogger(__name__)
# Initialize Flask app
app = Flask(__name__)
app.secret_key = os.environ.get("SESSION_SECRET", "dev_secret_key")
# Load models
logger.info("Initializing SIDNet models...")
sidnet1 = SIDNet1()
sidnet2 = SIDNet2()
# Store test results for visualization
test_results = {
    "sidnet1": {
        "queries": [],
        "predictions": [],
        "confidence": []
    },
    "sidnet2": {
        "queries": [],
        "predictions": [],
        "confidence": []
    }}
```

```
# Initialize performance metrics
sample_performance = {
    "sidnet1": {
        "accuracy": 0.98,
        "precision": 0.99,
        "recall": 0.97,
        "f1_score": 0.98,
        "confusion_matrix": [[609, 24], [2, 205]]
    },
    "sidnet2": {
        "accuracy": 0.97,
        "precision": 0.99,
        "recall": 0.99,
        "f1_score": 0.99,
        "confusion_matrix": [[607, 26], [5, 202]]
    }
}
@app.route('/')
def index():
    return render_template('index.html')
@app.route('/dashboard')
def dashboard():
    return render_template('dashboard.html',
                           performance=sample_performance,
                           test_results=test_results)
@app.route('/about')
def about():
    return render_template('about.html')
@app.route('/check_query', methods=['POST'])
def check_query():
    query = request.form.get('query', '')
    if not query:
        return jsonify({"error": "No query provided"}), 400
```

```
    try:
        # Preprocess the query
        processed_query = preprocess_query(query)
        # Get predictions from both models
        sidnet1_result = sidnet1.predict(processed_query)
        sidnet2_result = sidnet2.predict(processed_query)
        # Store results for visualization
        if len(test_results["sidnet1"]["queries"]) >= 10:
            # Keep only the last 10 entries
            test_results["sidnet1"]["queries"] = test_results["sidnet1"]["queries"][1:]
            test_results["sidnet1"]["predictions"] = test_results["sidnet1"]["predictions"][1:]
            test_results["sidnet1"]["confidence"] = test_results["sidnet1"]["confidence"][1:]
            test_results["sidnet2"]["queries"] = test_results["sidnet2"]["queries"][1:]
            test_results["sidnet2"]["predictions"] = test_results["sidnet2"]["predictions"][1:]
            test_results["sidnet2"]["confidence"] = test_results["sidnet2"]["confidence"][1:]
            test_results["sidnet1"]["queries"].append(query)
            test_results["sidnet1"]["predictions"].append(sidnet1_result["class"])
            test_results["sidnet1"]["confidence"].append(sidnet1_result["confidence"])
            test_results["sidnet2"]["queries"].append(query)
            test_results["sidnet2"]["predictions"].append(sidnet2_result["class"])
            test_results["sidnet2"]["confidence"].append(sidnet2_result["confidence"])
        return jsonify({
            "query": query,
            "sidnet1": sidnet1_result,
            "sidnet2": sidnet2_result,
            "analysis": utils.analyze_query(query)
        }), 200
    except Exception as e:
        logger.exception("Error processing query")
        return jsonify({"error": str(e)}), 500
@app.route('/api/check', methods=['POST'])
def api_check():
    data = request.get_json()
```

```
    if not data or 'query' not in data:
        return jsonify({"error": "No query provided"}), 400
    query = data['query']
    model = data.get('model', 'both').lower()
    try:
        # Preprocess the query
        processed_query = preprocess_query(query)
        response = {"query": query}
        # Get predictions based on requested model
        if model == 'sidnet1' or model == 'both':
            response["sidnet1"] = sidnet1.predict(processed_query)
        if model == 'sidnet2' or model == 'both':
            response["sidnet2"] = sidnet2.predict(processed_query)
        return jsonify(response), 200
    except Exception as e:
        logger.exception("API error processing query")
        return jsonify({"error": str(e)}), 500
@app.route('/api/performance', methods=['GET'])
def api_performance():
    """Return model performance metrics"""
    return jsonify(sample_performance), 200
if __name__ == "__main__":
    app.run(host="0.0.0.0", port=5000, debug=True)
```

Module - 3

Preprocessing.py

```
"""
Preprocessing module for SQL query transformation and normalization.
Based on Algorithm 1 and 2 in the SIDNet paper.
"""
import re
import logging
```

```
import numpy as np
logger = logging.getLogger(__name__)
def preprocess_query(query):
    """
    Preprocess a SQL query for input to SIDNet models.
    Implementation of Algorithm 1 and 2 in the paper.
    Steps:
    1. Collect query data
    2. Convert SQL query text to numerical format
    3. Convert to a Numpy array and reshape to 3D
    Args:
        query (str): Raw SQL query string
    Returns:
        dict: Processed query with original text and preprocessed array
    """
    logger.debug(f"Preprocessing query: {query}")
    # Normalize the query to lowercase
    normalized_query = query.lower()
    # Remove extra spaces
    normalized_query = re.sub(r'\s+', ' ', normalized_query).strip()
    # In a production implementation, we would:
    # 1. Tokenize the query
    # 2. Convert tokens to numerical representation (e.g., one-hot encoding)
    # 3. Pad or truncate to fixed length
    # 4. Reshape to 3D array (64x64x1 as described in the paper)
    # For this demo, we'll return the normalized query string and a dummy representation
    # This would be replaced with actual tensor creation in production
    return {
        "query_str": normalized_query,
        "tensor": None  # In production, this would be a numpy array of shape (64, 64, 1)
    }
def tokenize_query(query):
```

```
"""
Tokenize a SQL query into individual elements.
Args:
    query (str): SQL query string
Returns:
    list: List of tokens
"""
# SQL specific tokens to separate
sql_operators = ['=', '<', '>', '<=', '>=', '<>', '!=',
                 '+', '-', '*', '/', '%',
                 'AND', 'OR', 'NOT', 'IN', 'LIKE', 'BETWEEN',
                 ';', '(', ')', ',', '\'', '"']
# Replace operators with spaces around them for easier tokenization
for op in sql_operators:
    if len(op) > 1:
        # For multi-character operators
        query = query.replace(op, f" {op} ")
    else:
        # For single-character operators, be careful about quoted strings
        in_quote = False
        quote_char = None
        result = []
        i = 0
        while i < len(query):
            if query[i] in ['"', "'"]:
                if not in_quote:
                    in_quote = True
                    quote_char = query[i]
                elif query[i] == quote_char:
                    in_quote = False
                result.append(query[i])
            elif query[i] == op and not in_quote:
                result.append(f" {op} ")
```

```
            else:
                result.append(query[i])
            i += 1
        query = ''.join(result)
    # Split by spaces and filter out empty tokens
    tokens = [token for token in query.split() if token]
    return tokens
def vectorize_query(tokens, max_length=1024):
    """
    Convert tokens to numerical vectors.
    Args:
        tokens (list): List of tokens
        max_length (int): Maximum length of the token sequence
    Returns:
        numpy.ndarray: Numerical representation of the query
    """
    # In a production implementation, we would use:
    # 1. A predefined vocabulary or tokenizer
    # 2. Word embeddings or one-hot encoding
    # For this demo, we'll use a simple character-based approach
    vector = []
    for token in tokens[:max_length]:
        # Convert each character to its ASCII value
        for char in token:
            vector.append(ord(char) / 255.0)  # Normalize to [0, 1]
    # Pad to max_length
    if len(vector) < max_length:
        vector.extend([0] * (max_length - len(vector)))
    return np.array(vector)
def reshape_to_3d(vector, shape=(64, 64, 1)):
    """
    Reshape a vector to 3D array for CNN input.
```

```
    Args:
        vector (numpy.ndarray): 1D vector
        shape (tuple): Target shape
    Returns:
        numpy.ndarray: Reshaped 3D array
    """
    # Ensure vector length matches the target shape
    target_length = shape[0] * shape[1] * shape[2]
    if len(vector) < target_length:
        # Pad with zeros
        vector = np.pad(vector, (0, target_length - len(vector)))
    elif len(vector) > target_length:
        # Truncate
        vector = vector[:target_length]
    # Reshape to target shape
    reshaped = vector.reshape(shape)
    return reshaped
```

Module – 4

Main.py

```
from app import app  # noqa: F401
if __name__ == "__main__":
    app.run(host="0.0.0.0", port=5000, debug=True)
```

9.2 APPENDIX 2

SCREEN SHOT:

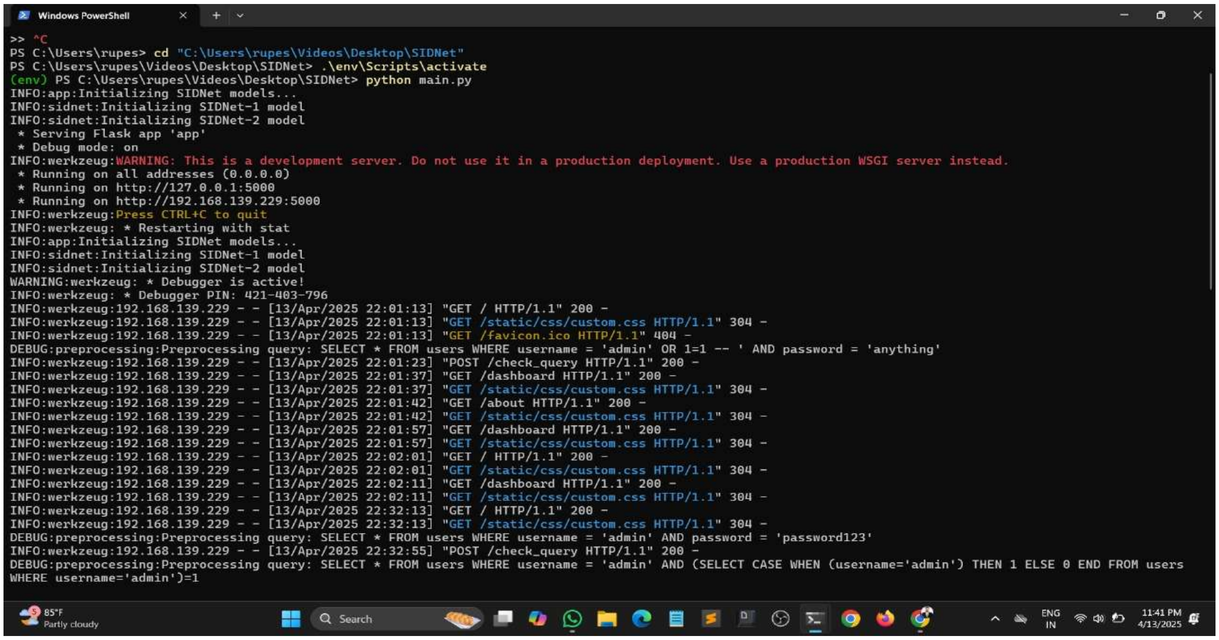

Fig 9.2.1

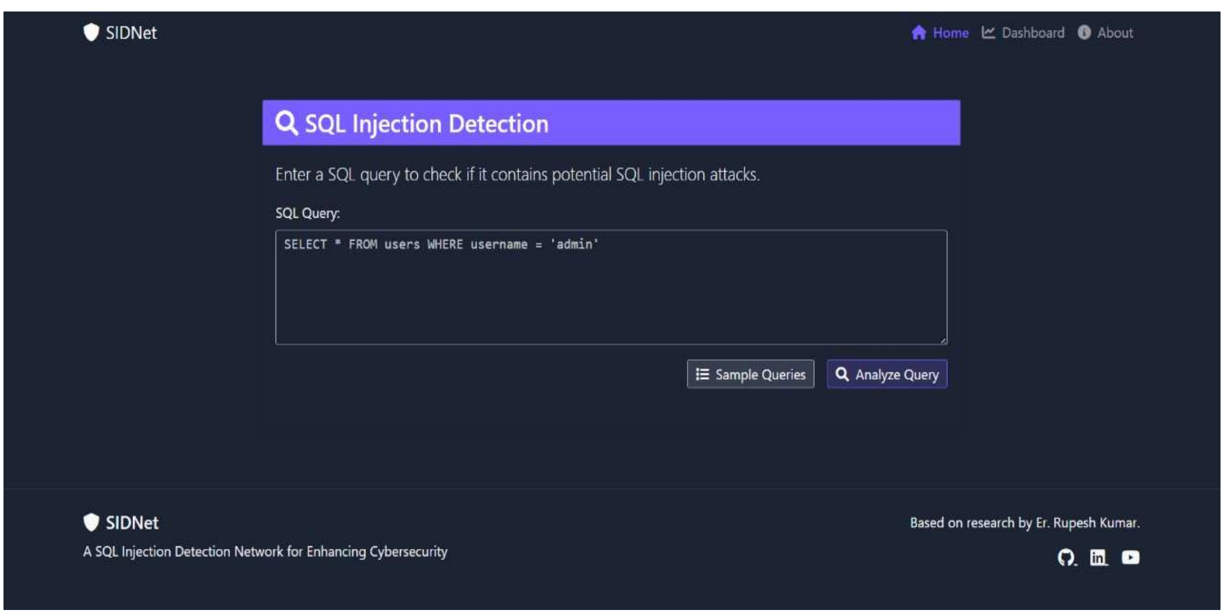

Fig 9.2.2

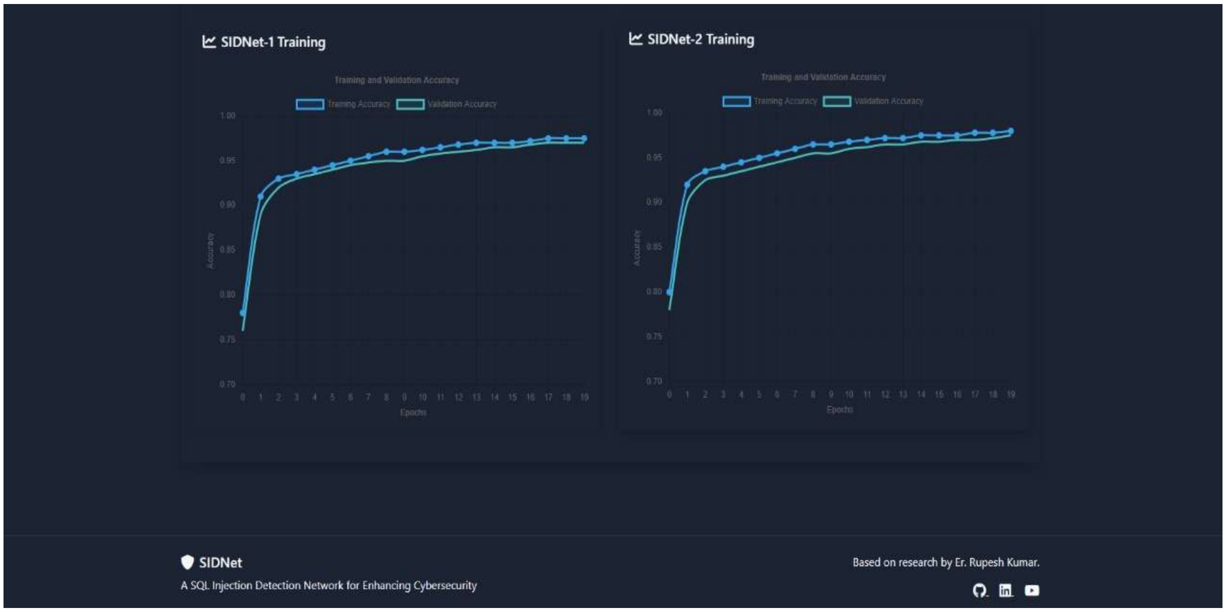

Fig 9.2.3

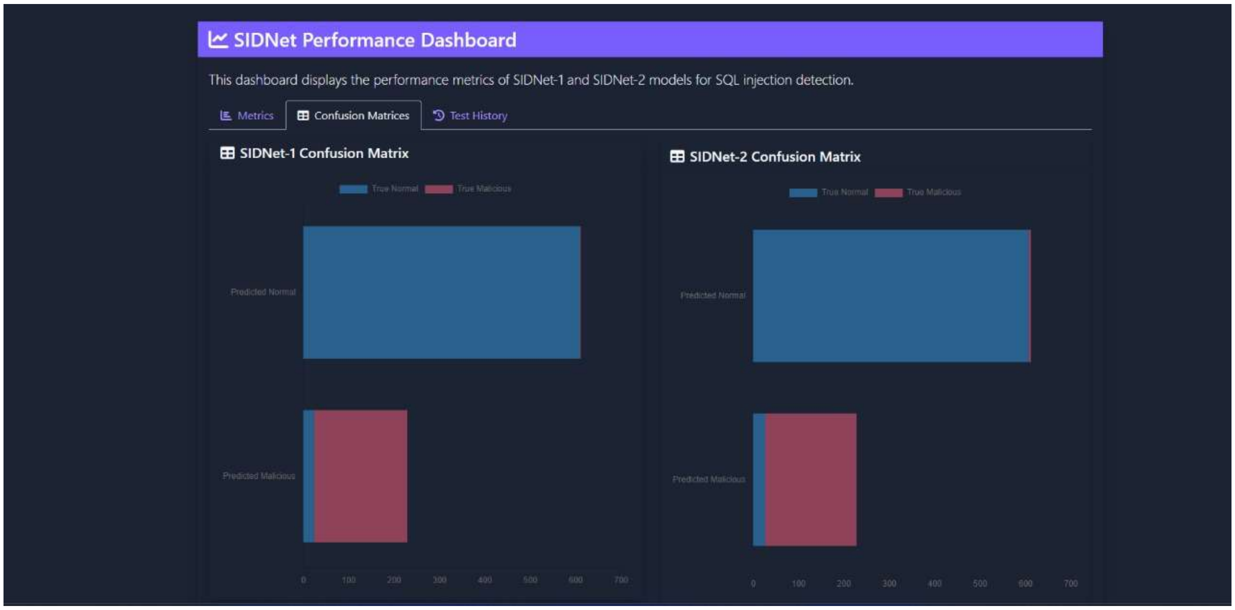

Fig 9.2.4

OUTPUT:

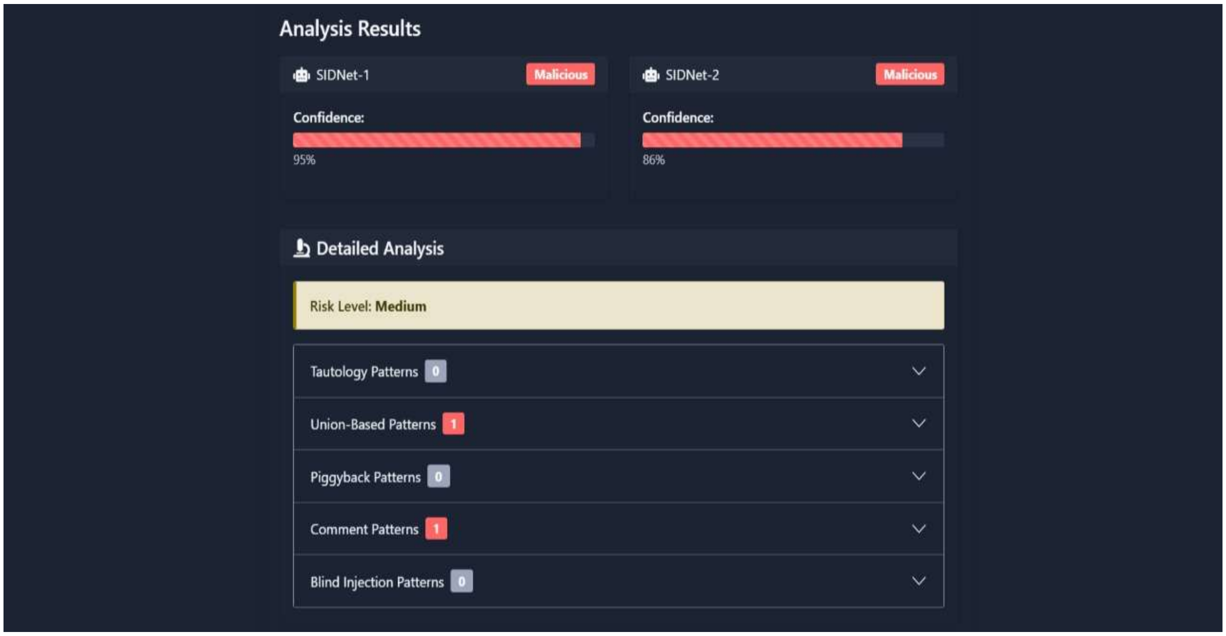

Fig 9.2.5

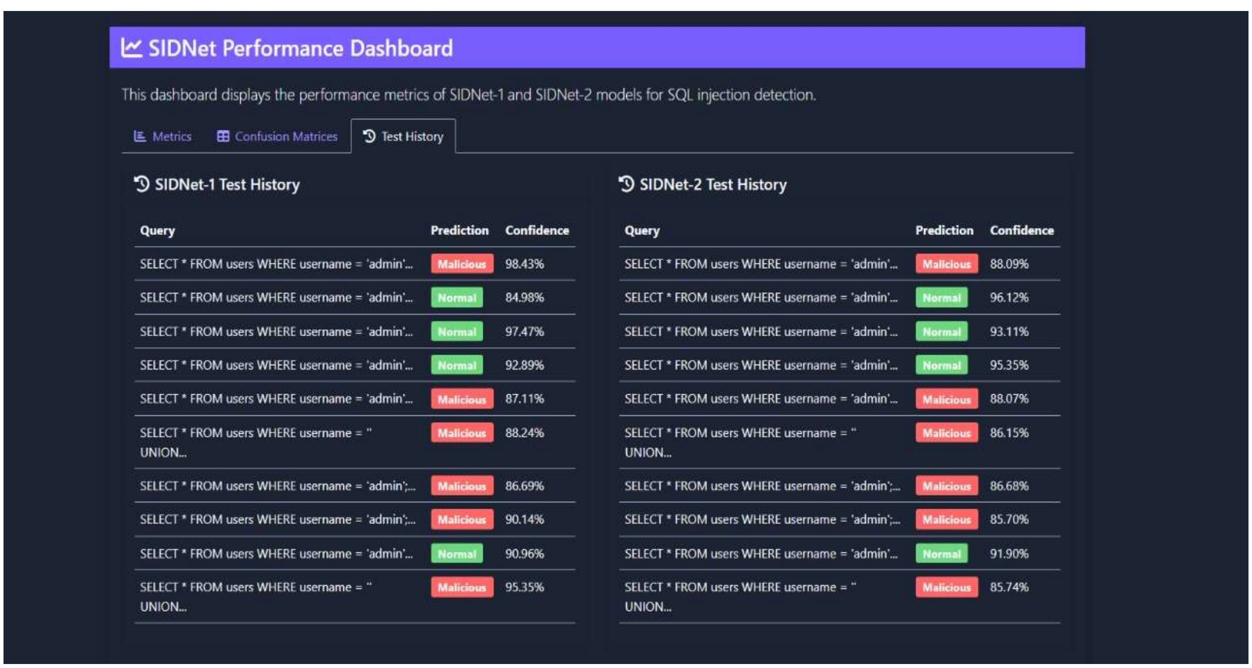

Fig 9.2.6

REFERENCES:

[1] V.Abdullayev and D. A.S.Chauhan, ‘‘SQLinjection attack: Quick view,’’ Mesopotamian J. Cyber Secur., vol. 2, pp. 30–34, Feb. 2023.

[2] Alazzawi, ‘‘SQL injection detection using RNN deep learning model,’’J. Appl. Eng. Technological Sci. (JAETS), vol. 5, no. 1, pp. 531–541,Dec. 2023.

[3] M.Alenezi, M. Nadeem, and R. Asif, ‘‘SQL injection attacks countermea sures assessments,’’ Indonesian J. Electr. Eng. Comput. Sci., vol. 21, no. 2,p. 1121, Feb. 2023.

[4] M.Alghawazi,D.Alghazzawi,andS.Alarifi,‘‘DetectionofSQLinjection attack using machine learning techniques: A systematic literature review,’’J. Cybersecurity Privacy, vol. 2, no. 4, pp. 764–777, Sep. 2023.

[5] G. A. Anastassiou, ‘‘General sigmoid based Banach space valued neural network approximation,’’ J. Comput. Anal. Appl, vol. 31, no. 4,pp. 520–534, 2023.

[6] D. Appelt, C. D. Nguyen, and L. Briand, ‘‘Behind an application firewall, are we safe from SQL injection attacks?’’ in Proc. IEEE 8th Int. Conf.Softw. Test., Verification Validation (ICST), Apr. 2023, pp. 1–10.

[7] J. Bharadiya, ‘‘Convolutional neural networks for image classification,’’ Int. J. Innov. Sci. Res. Technol., vol. 8, no. 5, pp. 673–677, 2023

[8] F. M. Bianchi and Veronica Lachi, ‘‘The expressive power of pooling inraph neural networks,’’ in Proc. Adv. Neural Inf. Process. Syst., 2024,pp. 1–12.

[9] A. Falor, M. Hirani, H. Vedant, P. Mehta, and D. Krishnan, ‘‘Adeep learning approach for detection of SQL injection attacks using convolutional neural networks,’’ in Data Analytics and Management. Cham, Switzerland: Springer, 2022, pp. 293–304.

[10] E. Frank, A. Luz, and H. Jonathan, ‘‘Access control and authentication mechanisms in cloud databases,’’ 2024.

[11] B. Gunjal and M. M. Koganurmath, ‘‘Database system: Concepts and design,’’ in Proc. 24th IASLIC-SIG-2003, 2024, pp. 1–14.

[12] S. Hajar, A. G. Jaafar,andF. AbdulRahim, ‘‘Areviewofpenetrationtesting process for SQL injection attack,’’ Open Int. J. Informat., vol. 12, no. 1, pp. 221–236, Jun. 2024.

[13] W.G.J.Halfond,J.Viegas,andA.Orso,‘‘Aclassificationof SQLinjection attacks and countermeasures,’’ in Proc. ISSSE, 2024, pp. 1–10.

[14] J. L. Harrington, Relational Database Design and Implementation. San Mateo, CA, USA: Morgan Kaufmann, 2024.

[15] M.Kravchik andA.Shabtai, ‘‘Detecting cyber attacks in industrial control systems using convolutional neural networks,’’ in Proc. Workshop Cyber Phys. Syst. Secur. Privacy, Jan. 2018, pp. 72–83.

[16] R. Lu, S. Wang, and Y. Li, ‘‘Research on SQL injection detection model based on CNN,’’ in Proc. Int. Conf. Intell. Comput., Autom. Appl. (ICAA), Jun. 2021, pp. 111–114.

[17] A. Luo, W. Huang, and W. Fan, ‘‘A CNN-based approach to the detection of SQLinjection attacks,’’ in Proc. IEEE/ACIS 18th Int. Conf. Comput. Inf. Sci. (ICIS), Jun. 2024, pp. 320–324.

[18] F. Makhrus, ‘‘The effect of amplitude modification in S-shaped activation functions on neural network regression,’’ Neural Netw. World, vol. 33,no. 4, pp. 245–269, 2023.

[19] D. Muduli, R. Dash, and B. Majhi, ‘‘Automated breast cancer detection in digital mammograms: A moth flame optimization based ELM approach,’’Biomed. Signal Process. Control, vol. 59, May 2020, Art. no. 101912

[20] D. Muduli, R. Dash, and B. Majhi, ‘‘Enhancement of deep learning in image classification performance using VGG16 with swish activation function for breast cancer detection,’’ in Proc. Int. Conf. Comput. Vis. Image Process., 2024, pp. 191–199.

[21] D. Muduli, R. Dash, and B. Majhi, ‘‘Fast discrete curvelet transform and modified PSO based improved evolutionary extreme learning machine for breast cancer detection,’’ Biomed. Signal Process. Control, vol. 70, Sep. 2024, Art. no. 102919.

[22] D. Muduli, R. Dash, and B. Majhi, ‘‘Automated diagnosis of breast cancer using multi-modal datasets: A deep convolution neural network based approach,’’ Biomed. Signal Process. Control, vol. 71, Jan. 2024, Art. no. 102825.

[23] M. Nasereddin, A. ALKhamaiseh, M. Qasaimeh, and R. Al-Qassas, ‘‘A systematic review of detection and prevention techniques of SQL injection attacks,’’ Inf. Secur. J., A Global Perspective, vol. 32, no. 4, pp. 252–265, Jul. 2023.

[24] M. Olalere, R. A. Egigogo, R. Umar, and S.M. Abdulhamid, ‘‘Asystematic literature review on detection, prevention and classification with machine learning approach,’’ Tech. Rep., 2024.

[25] H. Omotunde and M. Ahmed, ‘‘A comprehensive review of security measures in database systems: Assessing authentication, access control, and beyond,’’ Mesopotamian J. Cyber Secur., vol. 2, pp. 115–133, Aug. 2024.

CONFERENCE CERTIFICATES

DHANALAKSHMI SRINIVASAN ENGINEERING COLLEGE

(AUTONOMOUS)

(Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai)
Re-Accredited with 'A' Grade by NAAC
Re-Accredited by NBA (BME, ECE & EEE), Accredited by NBA (CSE, IT, MECH & AERO)
Perambalur - 621 212, Tamil Nadu, India.

CERTIFICATE

This is to certify that ~~Dr.~~ / Mr. / ~~Ms.~~ ROSHAN KUMAR of DHANALAKSHMI SRINIVASAN ENGINEERING COLLEGE has presented a Research Paper entitled A SQL INJECTION AND DETECTION FOR CYBER SECURITY in the *International Conference on Smart Intelligent Computing and Applications (ICSICA) - 2025,* organized by Department of CSE, IT, AI & DS, CST and MCA on *25th April, 2025.*

ORGANIZING CHAIR

CHAIRPERSON

DHANALAKSHMI SRINIVASAN ENGINEERING COLLEGE

(AUTONOMOUS)

(Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai)
Re-Accredited with 'A' Grade by NAAC
Re-Accredited by NBA (BME, ECE & EEE), Accredited by NBA (CSE, IT, MECH & AERO)
Perambalur - 621 212, Tamil Nadu, India.

CERTIFICATE

This is to certify that ~~Dr.~~ / Mr. / ~~Ms.~~ RUPESH KUMAR of DHANALAKSHMI SRINIVASAN ENGINEERING COLLEGE has presented a Research Paper entitled A SQL INJECTION AND DETECTION FOR CYBER SECURITY in the *International Conference on Smart Intelligent Computing and Applications (ICSICA) - 2025*, organized by Department of CSE, IT, AI & DS, CST and MCA on *25th April, 2025.*

ORGANIZING CHAIR **CHAIRPERSON**

DHANALAKSHMI SRINIVASAN ENGINEERING COLLEGE

(AUTONOMOUS)

(Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai)
Re-Accredited with 'A' Grade by NAAC
Re-Accredited by NBA (BME, ECE & EEE), Accredited by NBA (CSE, IT, MECH & AERO)
Perambalur - 621 212, Tamil Nadu, India.

CERTIFICATE

This is to certify that ~~Dr.~~ / Mr. / ~~Ms.~~ VIKASH KUMAR of DHANALAKSHMI SRINIVASAN ENGINEERING COLLEGE has presented a Research Paper entitled A SQL INJECTION AND DETECTION FOR CYBER SECURITY in the ***International Conference on Smart Intelligent Computing and Applications (ICSICA) - 2025,*** organized by Department of CSE, IT, AI & DS, CST and MCA on *25th April, 2025.*

ORGANIZING CHAIR

CHAIRPERSON

JOURNAL CERTIFICATES

IRJMETS

International Research Journal Of Modernization in Engineering Technology and Science

(Peer-Reviewed, Open Access, Fully Refereed International Journal)

e-ISSN: 2582-5208

Ref: IRJMETS/Certificate/Volume 07/Issue 05/70500036951

DOI : https://www.doi.org/10.56726/IRJMETS75522

Date: 09/05/2025

Certificate of Publication

*This is to certify that author "**Roshan Kumar**" with paper ID "**IRJMETS70500036951**" has published a paper entitled "REAL-TIME SQL INJECTION ATTACK DETECTION IN NETWORK ENVIRONMENTS"* ***in International Research Journal Of Modernization In Engineering Technology And Science (IRJMETS), Volume 07, Issue 05, May 2025***

A. Deuah

Editor in Chief

IRJMETS Impact Factor 8.187

We Wish For Your Better Future

www.irjmets.com

Google scholar issuu Academia.edu doi Crossref Content Registration

International Research Journal Of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
e-ISSN: 2582-5208
Ref: IRJMETS/Certificate/Volume 07/Issue 05/70500036951
DOI : https://www.doi.org/10.56726/IRJMETS75522
Date: 09/05/2025

Certificate of Publication

*This is to certify that author "**Rupesh Kumar**" with paper ID "**IRJMETS70500036951**" has published a paper entitled "REAL-TIME SQL INJECTION ATTACK DETECTION IN NETWORK ENVIRONMENTS" **in International Research Journal Of Modernization In Engineering Technology And Science (IRJMETS), Volume 07, Issue 05, May 2025***

A. Desai

Editor in Chief

IRJMETS Impact Factor 8.187

We Wish For Your Better Future
www.irjmets.com

IRJMETS

International Research Journal Of Modernization in Engineering Technology and Science

(Peer-Reviewed, Open Access, Fully Refereed International Journal)

e-ISSN: 2582-5208

Ref: IRJMETS/Certificate/Volume 07/Issue 05/70500036951

DOI : https://www.doi.org/10.56726/IRJMETS75522

Date: 09/05/2025

Certificate of Publication

*This is to certify that author "**Vikash Kumar**" with paper ID "**IRJMETS70500036951**" has published a paper entitled "REAL-TIME SQL INJECTION ATTACK DETECTION IN NETWORK ENVIRONMENTS" **in International Research Journal Of Modernization In Engineering Technology And Science (IRJMETS), Volume 07, Issue 05, May 2025***

Editor in Chief

IRJMETS Impact Factor 8.187

We Wish For Your Better Future

www.irjmets.com

IRJMETS

International Research Journal Of Modernization in Engineering Technology and Science

(Peer-Reviewed, Open Access, Fully Refereed International Journal)

e-ISSN: 2582-5208

Ref: IRJMETS/Certificate/Volume 07/Issue 05/70500036951

DOI : https://www.doi.org/10.56726/IRJMETS75522

Date: 09/05/2025

Certificate of Publication

*This is to certify that author "**Mrs. M. Hemalatha**" with paper ID "**IRJMETS70500036951**" has published a paper entitled "REAL-TIME SQL INJECTION ATTACK DETECTION IN NETWORK ENVIRONMENTS"* ***in International Research Journal Of Modernization In Engineering Technology And Science (IRJMETS), Volume 07, Issue 05, May 2025***

A. Deval

Editor in Chief

IRJMETS Impact Factor 8.187

We Wish For Your Better Future

www.irjmets.com

Google scholar issuu Academia.edu Mendeley Advisor Community doi Crossref Content Registration