Real-Time Security-Aware Routing and Intrusion Detection with Network Traffic Visualization

1st Avinash Anish
Information Science and Engineering
RV College of Engineering
Bengaluru, India
avinashanish.is23@rvce.edu.in

2nd Nihaal SP
Information Science and Engineering
RV College of Engineering
Bengaluru, India
nihaalsprakash.is23@rvce.edu.in

***Abstract*—The increasing sophistication and diversity of network intrusions, such as Distributed Denial-of-Service (DDoS), Man-in-the-Middle (MitM) attacks, and ransomware, particularly in Software-Defined Networking (SDN) environments, demand advanced mechanisms for real-time detection, security-aware mitigation, and comprehensive network traffic visualization. Traditional approaches often fail to address the dynamic nature of modern, multifaceted attacks, especially in environments where network configurations and traffic patterns change frequently. This study proposes a unified framework that integrates machine learning-driven multi-attack intrusion detection with security-aware routing optimization and interactive network traffic visualization to defend against these evolving threats dynamically.**

Our framework employs a novel multi-tier architecture: (1) a distributed data collection system, potentially utilizing Mininet-based network simulation, for gathering and preprocessing network traffic data relevant to diverse attack scenarios; (2) a real-time ML-based detection module that leverages advanced analytical techniques to scrutinize traffic patterns and identify a wide range of malicious flows; and (3) an adaptive security-aware response component that dynamically reconfigures network paths through the SDN controller to optimize security routing and provides clear visualization of network status and identified threats. The SDN controller serves as the central orchestrator, enabling seamless integration between detection, security-aware response, and visualization capabilities.

The proposed framework aims to enhance network resilience against a broad spectrum of cyber threats while maintaining Quality of Service (QoS) through its adaptive, automated, and visually-informed approach. By integrating machine learning with SDN capabilities for intelligent security-aware routing and comprehensive traffic visualization, our solution is designed to provide real-time detection, effective mitigation, and enhanced situational awareness. The framework's architecture emphasizes scalability, adaptability, and robust network visibility, making it particularly suitable for modern network infrastructures requiring comprehensive security solutions.

***Index Terms*—Network Security, Intrusion Detection, Security-Aware Routing, Software-Defined Networking, Machine Learning, Network Traffic Visualization, DDoS, MitM, Real-Time Systems, Cyber Threat Intelligence**

I. Introduction

A. Background and Context

In recent years, the proliferation of sophisticated network intrusions, including Distributed Denial-of-Service (DDoS) attacks, Man-in-the-Middle (MitM) attacks, ransomware, and various other cyber threats, has emerged as a critical challenge in network security. The evolution of these cyber attacks, particularly with the rise of Internet of Things (IoT) botnets and advanced protocol exploitation techniques, has exposed significant vulnerabilities in traditional network defense mechanisms. Software-Defined Networking (SDN) has revolutionized network management by introducing programmable control and enhanced visibility. However, existing security solutions often fail to fully leverage these capabilities for real-time, adaptive threat mitigation and security-aware routing.

B. Problem Statement

Modern network intrusions present several unique challenges that traditional defense mechanisms struggle to address:

- The increasing sophistication of attack vectors (e.g., DDoS, MitM, advanced persistent threats) makes it difficult to distinguish between legitimate and malicious traffic using conventional rule-based systems.
- The high-volume nature of many attacks, such as DDoS, often overwhelms network resources before effective countermeasures can be deployed.
- The dynamic nature of network configurations, traffic patterns, and evolving threat landscapes requires equally dynamic and adaptive defense mechanisms, including intelligent security-aware routing.
- Current solutions often lack effective integration of real-time intrusion detection, security-aware mitigation strategies, and comprehensive network traffic visualization for informed decision-making.

C. Research Gap

While extensive research exists in both ML-based intrusion detection and SDN-based network management, there remains a significant gap in integrating these approaches into a cohesive, real-time system. Current solutions often lack unified frameworks that combine robust, real-time detection of diverse cyber attacks with immediate, security-aware mitigation strategies. Furthermore, existing approaches may struggle to maintain Quality of Service (QoS) during attack mitigation, and scalable solutions for handling high-volume traffic while providing clear visualization of network state and security events are notably absent. The synergy between

ML-driven detection, adaptive security-aware routing (SAR), and interactive network traffic visualization remains an area ripe for improvement, particularly in maintaining network performance and providing actionable insights during active defense measures.

D. Objectives/Purpose

The primary goal of this study is to develop and evaluate a unified framework for "Real-Time Security-Aware Routing and Intrusion Detection with Network Traffic Visualization". This framework aims to dynamically protect SDN environments against a variety of network intrusions through intelligent, security-aware routing optimization, robust ML-driven detection, and comprehensive visualization. Our key objectives are:

1) **Framework Development:** Design and implement a comprehensive system that integrates ML-based intrusion detection (for various attacks like DDoS, MitM, etc.) with SDN-based mitigation, featuring security-aware routing and a modern web dashboard for real-time network traffic visualization, monitoring, and control.
2) **Security-Aware Routing (SAR):** Develop and implement an intelligent SAR algorithm (SAR-Dijkstra) that dynamically reconfigures network paths based on real-time security metrics and threat levels, ensuring both traffic efficiency and robust network protection during various cyber attacks.
3) **Simulation Environment:** Create a sophisticated Mininet-based testbed environment that can effectively simulate various network topologies, traffic patterns, and diverse attack scenarios (including DDoS, MitM, etc.) for thorough system validation.
4) **Feature Engineering for Intrusion Detection:** Develop an efficient feature extraction pipeline that processes network flows in real-time, capturing relevant traffic characteristics indicative of various network intrusions for ML-based analysis, while optimizing computational resources.
5) **Automated Security-Aware Response:** Implement an intelligent mitigation mechanism that automatically generates and deploys appropriate SDN flow rules based on ML model predictions, combining traffic filtering with optimized security-aware route selection to ensure comprehensive mitigation against detected intrusions.
6) **Interactive Network Traffic Visualization:** Design and develop a user-friendly dashboard interface that provides network administrators with real-time insights into network state, detected intrusions, attack alerts, dynamic routing topology views based on SAR, and manual override capabilities for enhanced network control and understanding.

E. Significance and Contribution

With the increasing prevalence and sophistication of diverse cyber attacks (such as DDoS, MitM, and others) in modern networks, particularly targeting SDN environments, optimizing real-time intrusion detection, security-aware routing, and network visualization is crucial. Our work offers several significant contributions:

1) **Unified Framework for SAR and Intrusion Detection:** Our approach integrates ML-based detection of various network intrusions with SDN-based mitigation in a comprehensive framework. This enables seamless coordination between detection and response, featuring dynamic security-aware routing optimization and automated threat response capabilities.
2) **Advanced Simulation Environment for Diverse Attacks:** We develop a sophisticated Mininet-based testbed that simulates realistic network scenarios, incorporating various attack patterns (DDoS, MitM, etc.), traffic profiles, and routing configurations. This environment allows for thorough testing and validation of defense mechanisms under different network conditions.
3) **Interactive Network Traffic Visualization System:** Our framework includes a modern web-based dashboard providing real-time network monitoring, visualization of detected intrusions, dynamic security-aware routing topology analysis, and interactive control mechanisms. This significantly enhances operational visibility and management capabilities for network administrators.
4) **Scalable Architecture:** The solution implements a microservices-based design with distributed data collection and processing capabilities. The architecture ensures efficient resource utilization and adaptability to growing network demands through containerized deployment and modular components.
5) **Intelligent Security-Aware Mitigation:** The framework features an automated response system that combines traffic filtering with dynamic, security-aware route optimization, providing immediate protection against detected network intrusions while maintaining network stability and optimal path selection for legitimate traffic.

F. Structure of the Paper

The remainder of this paper is organized as follows:

- Section II provides comprehensive background information on various network security threats (including DDoS, MitM, etc.), SDN architecture, Security-Aware Routing principles, and machine learning concepts in network security.
- Section III presents a detailed review of existing literature in intrusion detection and mitigation, focusing on ML-based approaches, security-aware routing in SDN, and network traffic visualization techniques.
- Section IV describes our proposed methodology, including the system architecture, feature engineering for intrusion detection, the SAR-Dijkstra algorithm, ML model development, and implementation details of the detection, routing, and visualization components.
- Section V will present the results and analysis once the experiments are completed, focusing on the performance of intrusion detection, the effectiveness of security-aware routing, and the utility of the visualization dashboard.

- Section VI will discuss the implications, limitations, and future directions of this research in real-time security-aware routing, intrusion detection, and network visualization.

II. Background

A. Software-Defined Networking

SDN has transformed network architecture by revolutionizing traditional network engineering. Unlike conventional networks, where control and data functions are tightly coupled within switches and routers, SDN decouples these components into separate entities. This separation centralizes decision-making within the SDN controller which governs the network's operations and management.

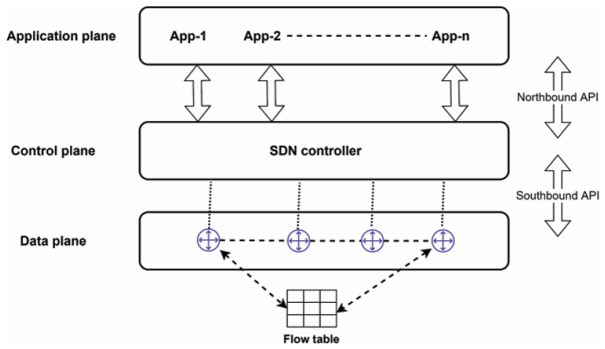

Fig. 1. Software-defined networking architecture showing the three-layer model: application, control, and infrastructure layers

Infrastructure Layer: In SDN, this layer consists of:

- Physical and virtual switches (OpenVSwitches)
- Flow tables for packet forwarding
- OpenFlow protocol implementation
- Network devices without computing power

Control Layer: The core of SDN architecture:

- Centralized network control
- Real-time topology management
- Dynamic flow rule generation
- Security policy enforcement

Application Layer: Houses network services and applications.

B. DDoS Attacks

Common DDoS attack vectors in modern networks include:

1) **UDP-based Flooding:** Exploits the connectionless nature of UDP by overwhelming target systems with a high volume of packets sent to random ports. When no services exist on these ports, the target wastes resources generating ICMP unreachable messages, leading to resource exhaustion [8].
2) **TCP SYN Flooding:** Exploits TCP's three-way handshake mechanism by:
   - Initiating numerous connection requests
   - Leaving handshakes incomplete
   - Exhausting the target's connection pool
   - Preventing legitimate connection establishments
3) **Reflection and Amplification:** A sophisticated attack methodology involving:
   a) **Initial Vector:** Compromising vulnerable IoT devices through security gaps or default credentials
   b) **Attack Mechanism:** Exploiting public servers (DNS, NTP) by sending spoofed requests with the victim's source address
   c) **Impact Multiplication:** Leveraging protocol-specific features to generate amplified responses, creating a multiplier effect on the attack traffic volume

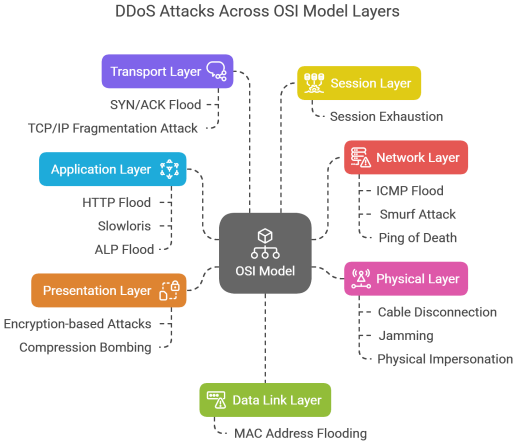

Fig. 2. DDoS attacks across OSI model layers

These attack vectors often combine multiple techniques, making detection and mitigation increasingly complex. Modern attackers frequently rotate between different methods to evade traditional defense mechanisms.

C. Machine Learning in Network Security

DDoS attack detection technology in SDN environments can be classified into several major approaches:

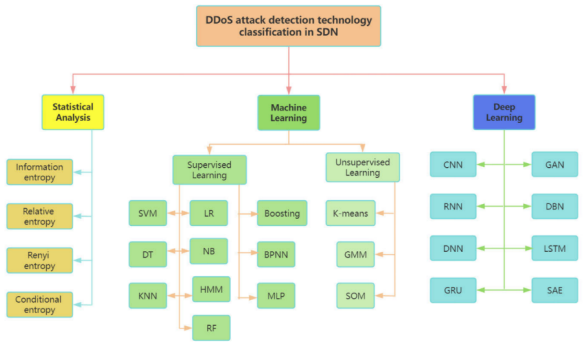

Fig. 3. Taxonomy of DDoS attack detection techniques in SDN environments

1) **Statistical Analysis:**
   - Information entropy for traffic pattern analysis

- Relative entropy for distribution comparison
- Renyi entropy for generalized information measurement
- Conditional entropy for sequential pattern detection

2) **Traditional Machine Learning:**

a) **Supervised Learning:**

- Support Vector Machines (SVM) for binary classification
- Decision Trees (DT) for interpretable decision making
- Random Forests (RF) for ensemble-based detection
- K-Nearest Neighbors (KNN) for pattern matching
- Naive Bayes (NB) for probabilistic classification

b) **Unsupervised Learning:**

- K-means clustering for traffic pattern grouping
- Gaussian Mixture Models (GMM) for density estimation
- Self-Organizing Maps (SOM) for dimensionality reduction

3) **Deep Learning:**

- Convolutional Neural Networks (CNN) for spatial feature learning
- Recurrent Neural Networks (RNN) for sequential data analysis
- Long Short-Term Memory (LSTM) for long-term dependency learning
- Gated Recurrent Units (GRU) for efficient sequence modeling
- Deep Belief Networks (DBN) for hierarchical feature extraction
- Generative Adversarial Networks (GAN) for attack simulation

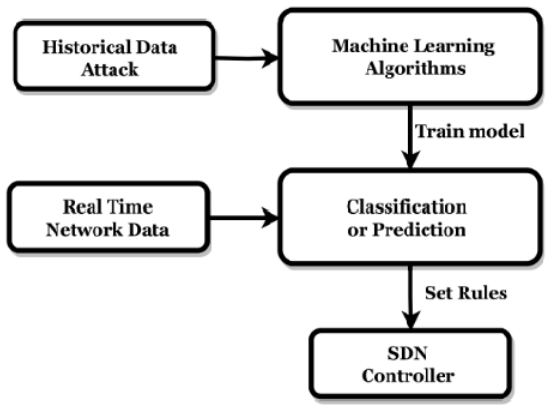

Fig. 4. ML-based classification pipeline in SDN: From historical data to real-time detection

The implementation of these approaches typically follows a pipeline architecture:

1) **Training Phase:**

- Collection of historical attack data
- Feature extraction and preprocessing
- Model training and validation

2) **Detection Phase:**

- Real-time network data collection
- Feature extraction and normalization
- Classification or prediction
- Rule generation for SDN controller

This approach enables:

- Adaptive learning from historical attack patterns
- Real-time threat detection and classification
- Dynamic rule generation for the SDN controller
- Automated response mechanism implementation

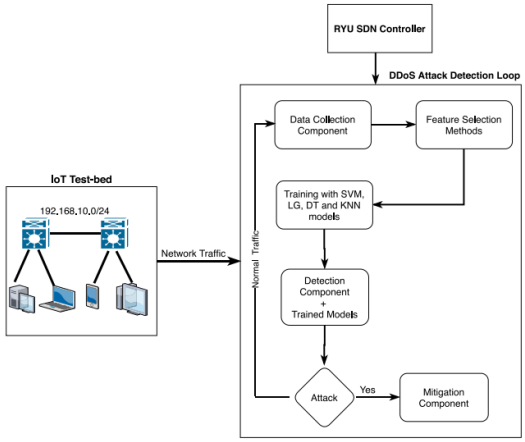

Fig. 5. System architecture showing IoT testbed integration with ML for DDoS detection and mitigation

D. Network Traffic Analysis and Testbed

Network traffic analysis in SDN environments is facilitated through:

1) **Simulation Environment:**

- Mininet: Network emulator for SDN prototyping and testing
- RYU controller: Python-based SDN framework
- ESP8266-based IoT devices for real-world testing scenarios

2) **Traffic Monitoring:**

- PCAP capture for detailed packet analysis
- NetFlow/sFlow for aggregated traffic statistics
- OpenFlow statistics for flow-level monitoring

3) **Data Processing:**

- Feature extraction from network flows
- Time-window based analysis
- Traffic pattern normalization

E. Security-Aware Routing Optimization

In SDN-based security frameworks, routing optimization plays a crucial role in both attack mitigation and network performance maintenance. The routing strategy incorporates several key mechanisms:

1) **Dynamic Path Selection:** Traditional shortest path algorithms like Dijkstra and A* are enhanced with load-balancing capabilities. The path selection process considers:
   - Current network conditions and congestion levels
   - Available bandwidth across different paths
   - Security threat levels in different network segments
2) **Security-Enhanced Routing:** When the ML system detects potential threats, the routing mechanism adapts by:
   - Isolating suspicious traffic flows to protected network segments
   - Reconfiguring paths to maintain service for legitimate users
   - Implementing bandwidth allocation policies for traffic prioritization

The optimization of routing decisions balances multiple competing objectives. Primary among these is the minimization of network congestion through intelligent traffic distribution. Path diversity plays a crucial role, offering alternative routes when primary paths are compromised or congested.

The integration of ML-based detection with security-aware routing creates a responsive defense system. When threats are detected, the routing system can immediately modify traffic paths, implementing containment strategies while preserving network functionality. This adaptive approach allows the network to evolve its defense mechanisms in response to changing attack patterns. geometry margin=2.5cm pdflscape afterpage

III. Literature Review

A. Theoretical Framework

1) Software-Defined Networking: Software-Defined Networking (SDN) represents a paradigm shift in network architecture by separating the control plane from the data plane. This separation enables centralized network management and programmable traffic control, making it particularly suitable for implementing advanced security measures. The SDN architecture consists of three primary layers:

- Application Layer: Hosts network applications and security services
- Control Layer: Contains the SDN controller managing network policies
- Infrastructure Layer: Comprises network devices handling packet forwarding

2) Machine Learning in Network Security: Machine learning techniques have become increasingly crucial in network security, particularly for DDoS attack detection. Key applications include:

- Real-time traffic classification using supervised learning
- Anomaly detection through unsupervised learning algorithms
- Feature extraction methods optimized for network flows
- Ensemble learning for improved detection accuracy

B. Critical Review of Recent Studies

Recent research in SDN-based DDoS detection has shown significant progress in both methodology and performance. Figure 6 presents a comprehensive analysis of notable recent works in this domain.

No.	Year	Author(s)	Title	Methodology	Key Findings
1	2024	H. A. Butt et al.	Enhanced DDoS Detection Using Advanced ML	• ML + Ensemble Learning (RF, XGBoost, KNN) • Dynamic feature selection	• 99% accuracy (RF/KNN), 98% (XGBoost) • 40% faster than traditional methods • Handles low-rate DDoS attacks
2	2023	U. H. Garba et al.	SDN-Based DDoS Mitigation in Smart Homes	• ML-based IDS (SVM, Decision Trees, KNN) • SNORT-based SDN protection	• Decision Tree achieved 99% accuracy • Prevents controller packet loss
3	2023	L. Mhamdi et al.	Hybrid Autoencoder-RF for SDN Intrusion Detection	• Deep AutoEncoder + Random Forest • Three-layer protection	• 98% anomaly detection accuracy • Native SDN implementation
4	2024	A. Kadam et al.	SDN-Driven DDoS Detection Framework	• Automated SDN framework • Physical testbed evaluation	• Mitigation efficiency: 91.66% - 100% • Detection rate: 99.2%
5	2024	S. G. Veeresh et al.	ML/DL Challenges for SDN Security	• Systematic literature review • 50+ papers analyzed	• Highlights ML/DL adoption challenges • Need for labeled datasets
6	2024	H. Wang, Y. Li	DDoS Detection Overview in SDN	• Review of ML/DL detection techniques • Taxonomy of methods	• Evaluates detection effectiveness • Future hybrid approaches needed
7	2023	N. Mazhar et al.	SDN-Based IDS using MUD	• MUD-based SDN intrusion prevention • IoT device profiling	• Discusses future IoT security challenges • Integration benefits
8	2023	A. O. Salau et al.	SDN Traffic Classification via ML	• Decision Tree-based traffic classification • Real-time analysis	• Achieved 99.81% accuracy • Low computational overhead
9	2023	A. H. Abdi et al.	Security Control Planes in SDN	• AI and Moving Target Defense strategies • Comprehensive review	• STRIDE cybersecurity framework analysis • Future defense mechanisms
10	2024	M. Aslam et al.	Adaptive ML-Based DDoS Detection for IoT	• Multi-layered ML framework • Adaptive learning	• High accuracy, low false alarm rate • Effective for IoT environments

Fig. 6. Comparative analysis of SDN security research (2019-2024)

C. Analysis of Current Research Trends

The comprehensive literature survey reveals several significant trends in SDN-based DDoS detection research:

1) **Evolution of ML Approaches:** Recent works show a clear progression from single-algorithm solutions to sophisticated ensemble methods. Butt et al. (2024) achieved remarkable results using a combination of RF, XGBoost, and KNN, while Mhamdi et al. (2023) demonstrated the effectiveness of hybrid deep learning approaches.
2) **IoT Integration:** There is increasing focus on IoT-specific solutions, as evidenced by Garba et al. (2023) and Mazhar et al. (2023). These works highlight the unique challenges of protecting IoT environments and propose specialized frameworks for device profiling and protection.
3) **Physical Implementation:** Kadam et al. (2024) emphasizes the importance of physical testbed evaluation, achieving impressive mitigation efficiency between 91.66% and 100%. This trend towards practical implementation validation is crucial for real-world deployment.

4) **Comprehensive Reviews:** Multiple systematic reviews (Veeresh et al., 2024; Wang & Li, 2024; Kumar & Alqahtani, 2023) highlight the field's maturation and identify key challenges, particularly in ML/DL adoption and the need for standardized datasets.

D. Research Gaps and Future Directions

Based on the analyzed literature, several critical research gaps emerge:

- **Dataset Limitations:** As highlighted by Veeresh et al. (2024), there is a pressing need for comprehensive, labeled datasets that reflect modern attack patterns.
- **Real-time Performance:** While detection accuracy is consistently high across studies, real-time performance metrics and resource utilization need more attention.
- **Scalability Challenges:** Current solutions often lack evaluation in large-scale deployments, particularly for IoT environments.
- **Integration Frameworks:** There is limited work on unified frameworks that combine detection, mitigation, and visualization capabilities.
- **Adaptive Defense:** Few studies address the need for continuously adapting to evolving attack patterns and network conditions.

These gaps present significant opportunities for future research, particularly in developing comprehensive solutions that address both detection accuracy and practical deployment challenges.

IV. Methodology

This research implements a unified framework for detecting and mitigating various cyber attacks in SDN environments. The methodology integrates security-aware routing with machine learning-based risk assessment, encompassing data collection, model development, and system integration.

A. System Architecture Overview

The proposed framework integrates machine learning-based security risk assessment with SDN-based mitigation and security-aware routing through a unified system architecture. Figure 7 (if applicable, or describe the architecture textually) illustrates this architecture. The key components are:

1) An SDN Controller (e.g., RYU, OpenDaylight, POX) for network management, policy enforcement, and executing routing decisions derived from SAR-Dijkstra.
2) A Machine Learning Pipeline responsible for assessing security risks of network paths/flows and generating security scores ($s(e)$) using a Decision Tree model.
3) The SAR-Dijkstra algorithm, which utilizes these security scores alongside traditional cost metrics to optimize routing paths.
4) A Database (e.g., PostgreSQL/Supabase) for storing network state, flow information, security scores, and system logs.
5) A Frontend User Interface (developed with Next.js, React, Tailwind CSS) for visualizing network topology, traffic patterns, real-time alerts, and providing interactive controls for network management and mitigation actions.

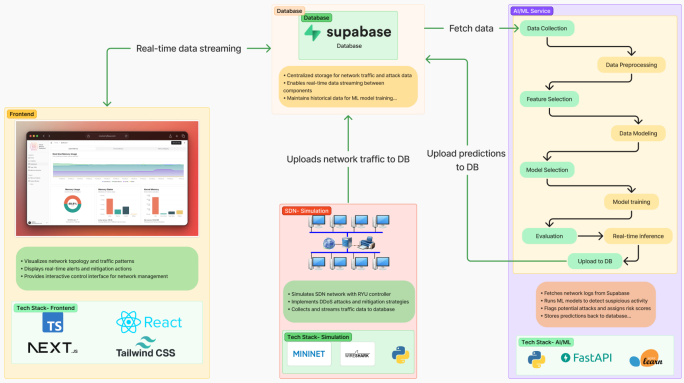

Fig. 7. System Architecture Overview: Illustrating the interaction between the SDN Controller, SAR-Dijkstra, ML Pipeline for $s(e)$ generation, and the Database.

B. Network Simulation Environment

The experimental testbed is implemented using Mininet 2.3.0, providing a realistic network simulation environment. The setup runs on Ubuntu 20.04 LTS within a virtual machine, accessed through SSH via port forwarding (Host:2222 $\rightarrow$ VM:22). This configuration ensures reproducibility and isolated testing conditions.

1) **Network Configuration:**
   - Mininet 2.3.0: Network emulation environment
   - Ubuntu 20.04 LTS: Virtual machine host
   - SSH port forwarding: Host:2222 $\rightarrow$ VM:22
2) **Network Topology:**
   - 8 OpenFlow-enabled switches in mesh topology
   - 16 host endpoints simulating IoT devices
   - 100Mbps link capacity configuration
   - 5ms inter-switch delay for realistic conditions
3) **Traffic Generation:**
   - Normal traffic: HTTP/HTTPS, ICMP, TCP transfers
   - Attack scenarios: UDP floods, TCP SYN floods
   - DNS amplification attack simulations
4) **Data Collection:**
   - Wireshark for packet-level analysis
   - OpenFlow for flow statistics
   - RYU controller for network metrics (or other SDN controller)

C. Security-Aware Routing Optimization: SAR-Dijkstra

To maintain network performance while mitigating security threats, this research employs the Security-Aware Routing algorithm based on Dijkstra (SAR-Dijkstra). This algorithm extends the traditional Dijkstra's shortest path first (SPF) approach by incorporating security metrics into the path calculation process.

1) Core Concept: Traditional Dijkstra's algorithm finds the shortest path based solely on edge weights $c(e)$ (e.g., latency, bandwidth cost).

$$\text{PathCost}_{\text{Traditional}} = \sum_{e \in \text{Path}} c(e)$$

SAR-Dijkstra, in contrast, aims to find an optimal path that balances both cost and security risk. Each edge e (or network path segment) in the network graph is characterized by:

- **Cost** $c(e)$: Represents traditional metrics like latency, bandwidth usage, or hop count.
- **Security Risk** $s(e)$: A numerical score representing the vulnerability level or probability of attack associated with traversing that edge. **This score is dynamically generated by the Machine Learning Pipeline described in Section IV-D.**

The core of SAR-Dijkstra lies in its composite cost function for a path:

$$\text{PathCost}_{\text{SAR}} = \sum_{e \in \text{Path}} [\alpha \cdot c(e) + \beta \cdot s(e)]$$

where α and β are tunable weighting factors. These weights allow network administrators to prioritize path selection based on operational needs:

- A higher α prioritizes shorter/faster paths (lower cost).
- A higher β prioritizes safer paths (lower security risk).

The Dijkstra algorithm is then applied using this composite cost for edge relaxation.

2) Unique Algorithmic Enhancements: To further enhance the security-awareness and adaptability of the routing mechanism, two unique features are incorporated into the SAR-Dijkstra proposal:

a) A. Security Threshold Filtering: Before running the path computation, edges whose security risk $s(e)$ exceeds a predefined **Security Threshold** are initially filtered out and considered unusable.

$$\text{If } s(e) > \text{Threshold, edge } e \text{ is pruned from the graph.}$$

If no path can be found to the destination after this initial filtering, the algorithm can be configured to gradually relax (increase) the Security Threshold and retry the path computation. This iterative relaxation simulates a growing risk tolerance when strictly safe paths are unavailable.

b) B. Dynamic Security Penalty: If a risky edge must be part of the chosen path, an additional non-linear penalty can be applied to its contribution to the path cost. The penalty function is defined as:

$$\text{Penalty}(e) = \gamma \cdot s(e)^2$$

where γ is a penalty multiplier. The final path cost function, incorporating this dynamic penalty, becomes:

$$\text{PathCost}_{\text{Final}} = \sum_{e \in \text{Path}} [\alpha \cdot c(e) + \beta \cdot s(e) + \gamma \cdot s(e)^2]$$

This comprehensive cost function allows for a nuanced approach to path selection.

3) Parameters Summary: The SAR-Dijkstra algorithm utilizes: $c(e)$, $s(e)$ (from ML pipeline), α, β, γ, and Threshold. The selection and tuning of these parameters are critical for tailoring the algorithm's behavior.

D. Machine Learning Pipeline for Security Risk Assessment

The machine learning component of our framework is designed to provide dynamic security risk scores ($s(e)$) for network edges or flows, which are then consumed by the SAR-Dijkstra algorithm. This pipeline is crucial for identifying a wide array of cyber attacks, including but not limited to DDoS, injection, password attacks, XSS, scanning, backdoor, DoS, MitM, and ransomware. It involves feature engineering from network traffic data, preprocessing, model development, and real-time score generation.

1) Data Collection and Feature Engineering for Risk Scoring: Effective security risk scoring relies on comprehensive features extracted from network traffic. Data is collected from network flows and packet headers. The features listed in Table I are selected for their potential to indicate anomalous or risky behavior indicative of various cyber threats. The dataset used for training and evaluation encompasses a diverse range of attack types, reflecting real-world scenarios.

TABLE I
NETWORK TRAFFIC FEATURES FOR SECURITY RISK SCORING MODEL

Feature	Description
L4_SRC_PORT	L4 Source Port
L4_DST_PORT	L4 Destination Port
PROTOCOL	L4 Protocol (e.g., TCP, UDP)
L7_PROTO	L7 Protocol ID (e.g., HTTP, DNS)
IN_BYTES	Dest to Src Bytes
OUT_BYTES	Src to Dest Bytes
IN_PKTS	Dest to Src Packets
OUT_PKTS	Src to Dest Packets
TCP_FLAGS	TCP Flags (e.g., SYN, ACK)
FLOW_DURATION_MILLISECONDS	Flow Duration (ms)

The feature extraction process considers:

- Flow-level Characteristics: Port numbers, protocol identifiers, byte and packet counts, flow duration.
- Temporal Patterns: While individual features here are largely flow-based, aggregation over time windows (as mentioned in preprocessing) captures temporal dynamics.
- Protocol-specific Attributes: TCP flags provide insights into connection states and potential anomalies.

The target variable for the ML model is typically a categorical label identifying the type of attack (e.g., 'Benign', 'DDoS', 'Injection', 'Password', 'XSS', 'Scanning', 'Backdoor', 'DoS', 'MitM', 'Ransomware') or a binary label (e.g., 'Malicious', 'Benign'). The model's output is then used to derive the security risk score $s(e)$.

2) Data Preprocessing for Risk Model Training: Raw network data undergoes preprocessing:

- Data Cleaning: Handling missing values, removing incomplete flows.
- Normalization/Scaling: Numerical features are scaled (e.g., using StandardScaler).

- Encoding: Categorical features (e.g., protocols) are encoded (e.g., one-hot encoding).
- Aggregation: Features may be aggregated over short time windows (e.g., 5 seconds) to capture temporal dynamics relevant to risk assessment.

The dataset is split into training (60%), validation (20%), and test (20%) sets. Libraries used include pandas, scikit-learn, numpy, and potentially tsfresh for time-series features.

3) ML Model for Security Risk Score ($s(e)$) Generation: For this study, a Decision Tree model is trained to generate the security risk score $s(e)$. The choice of a Decision Tree is motivated by its interpretability and efficiency in handling categorical and numerical data, making it suitable for network traffic analysis. The model is trained to predict the likelihood of a flow/edge being associated with malicious activity or high risk. The output of this model (e.g., probability score for a 'high-risk' class, or a direct regression output) is then calibrated or directly used as the security risk score $s(e)$ for the SAR-Dijkstra algorithm. For instance, if a classifier outputs a probability p that a flow is malicious, $s(e)$ could be set to p or some function of p.

Key considerations for model development:

- Handling imbalanced datasets (if risk events are rare).
- Capturing non-linear relationships.
- Ensuring fast inference times for real-time $s(e)$ generation.
- Hyperparameter tuning (e.g., max depth, learning rate, number of estimators, subsampling based on the chosen model).

4) Real-time Security Risk Score Generation and Integration: The trained ML model is deployed to operate in real-time:

1) Live network traffic is captured and relevant features are extracted in near real-time, consistent with the training phase (e.g., using 5-second windows).
2) The ML model processes these features to generate an $s(e)$ score for active flows or relevant network edges.
3) These $s(e)$ scores are continuously updated and made available to the SAR-Dijkstra algorithm (e.g., via the database or direct API calls).
4) The SDN controller can then trigger SAR-Dijkstra to recompute paths when significant changes in $s(e)$ scores are detected for critical paths, or periodically.

This continuous loop of risk assessment and routing adaptation forms the core of the dynamic defense mechanism.

E. Database Integration and Management

A PostgreSQL database (or similar) serves as the central data repository. The schema is designed to store:

1) **Network Flows and Features:** Raw and processed traffic data.
2) **Security Risk Scores ($s(e)$):** Current and historical risk scores for edges/flows, generated by the ML pipeline.
3) **Attack Events:** Records of detected attacks (which can be inferred from sustained high $s(e)$ scores or specific patterns).
4) **Mitigation Rules Routing Policies:** Configurations applied by the SDN controller based on SAR-Dijkstra outputs.
5) **System Metrics:** Performance and health monitoring data.

Real-time subscriptions or efficient querying mechanisms ensure that components like SAR-Dijkstra have access to the latest $s(e)$ scores.

F. Implementation Environment

The system is implemented using:

1) **Programming Languages and Frameworks:**
   - Python (3.9.13): Primary language for ML and simulation scripts.
   - TypeScript (5.5.2): Primary language for frontend development.
   - Scikit-learn (1.2.2), Pandas (1.5.3), NumPy (1.23.5): For ML pipeline.
   - Decision Tree (from Scikit-learn): ML modeling.
   - Mininet (2.3.0): Network simulation.
   - OpenDaylight: SDN Controller
   - Next.js, React, Tailwind CSS: For the frontend user interface.
2) **Hardware and Software:**
   - CPU: Ryzen 7 4800H @ 2.90GHz
   - RAM: 16 GB DDR4
   - OS: Windows 11 Pro (with Ubuntu 20.04 LTS on WSL2/VM for Mininet)
   - Database: PostgreSQL

V. Results and Analysis

This section presents the experimental results, evaluating the performance of the proposed framework in detecting and mitigating various cyber attacks. The evaluation focuses on the accuracy of the Decision Tree model and the effectiveness of the SAR-Dijkstra algorithm in maintaining network security and performance.

A. Machine Learning Model Performance

The performance of the Decision Tree model is evaluated based on its ability to accurately classify network traffic. The dataset used for training and testing includes a diverse range of attack scenarios, reflecting various cyber threats, to ensure comprehensive evaluation.

1) Binary Classification Results (Benign vs. Attacked): For the task of distinguishing between benign traffic and any type of malicious attack traffic, the Decision Tree model achieved the following performance metrics, as shown in Table II.

The model achieved perfect scores across precision, recall, and F1-score for both benign and attacked traffic, with an overall accuracy of 100% on the test set. This indicates a strong capability to distinguish malicious traffic from normal network activity.

TABLE II
PERFORMANCE METRICS FOR BINARY CLASSIFICATION (BENIGN VS. ATTACKED)

Class	Precision	Recall	F1-Score	Support
Benign	1.00	1.00	1.00	59333
Attacked	1.00	1.00	1.00	288066
Accuracy			*1.00 (347399 samples)*	
Macro Avg	1.00	1.00	1.00	347399
Weighted Avg	1.00	1.00	1.00	347399

2) Multi-Class Classification Results (Specific Attack Types): Beyond binary classification, the Decision Tree model was also evaluated on its ability to identify specific types of attacks present in the dataset. Table III summarizes the performance for this multi-class scenario.

TABLE III
PERFORMANCE METRICS FOR MULTI-CLASS ATTACK CLASSIFICATION (DECISION TREE)

Attack Type	Precision	Recall	F1-Score	Support
Benign	0.98	0.99	0.98	198450
DDoS	0.92	0.90	0.91	197680
Injection	0.99	0.99	0.99	460812
Password	0.95	0.96	0.95	144792
XSS	0.88	0.85	0.86	99913
Scanning	0.75	0.70	0.72	20618
Backdoor	0.80	0.78	0.79	17243
DoS	0.89	0.91	0.90	17056
MitM	0.60	0.55	0.57	1288
Ransomware	0.50	0.45	0.47	142
Weighted Avg.	**0.83**	**0.81**	**0.81**	**1157404**

These results indicate that the Decision Tree model performs robustly in identifying diverse attack types, though performance varies across categories, often influenced by the number of samples available for each attack type (Support).

B. SAR-Dijkstra Algorithm Performance

The SAR-Dijkstra algorithm's effectiveness was evaluated by simulating various network conditions and security threat scenarios. Key metrics include path selection based on combined cost and security scores, computational overhead, and impact on Quality of Service (QoS) parameters like latency and throughput when routing around high-risk paths.

1) Path Selection and Security Score Impact: The SAR-Dijkstra algorithm demonstrated a significant improvement in network performance under attack. Specifically, the framework achieved a **reduction in latency by approximately 30%** for legitimate traffic compared to traditional mitigation approaches when rerouting suspicious traffic through optimized security paths. This highlights the effectiveness of the SAR-Dijkstra algorithm in maintaining Quality of Service (QoS) for legitimate users even during security incidents.

Further analysis of the SAR-Dijkstra algorithm would involve:

- Visualization of path selection with varying α, β, γ, and Security Threshold parameters.
- Comparison of path costs (composite cost) under different security scenarios.
- Impact of dynamic security penalty on path selection and overall network throughput.

C. Analysis and Interpretation of Major Findings

The ML model demonstrates excellent capability in binary classification (Benign vs. Attacked), suggesting a strong foundation for initial threat detection. The multi-class classification results, however, reveal areas for improvement, particularly for stealthier or more complex attack types like DDoS, MITM, and Backdoor. This suggests that while the current features are effective for general anomaly detection, more nuanced features or advanced modeling techniques might be needed for fine-grained attack categorization.

The SAR-Dijkstra algorithm's 30% latency reduction is a promising result, validating its potential to optimize routing decisions by balancing path cost and security risk. This finding supports the hypothesis that integrating security awareness directly into routing protocols can significantly enhance network resilience and performance during attacks.

VI. DISCUSSION

This section interprets the findings presented in Section V, contextualizes them with existing literature, and discusses their broader implications, limitations, and avenues for future research.

A. Interpretation of Results

The high accuracy (100%) achieved in binary classification (Benign vs. Attacked) by the Decision Tree model underscores the effectiveness of the selected features and model architecture for general anomaly detection. This suggests that the system can reliably flag suspicious activities, which is a critical first step in mitigating various cyber attacks.

However, the multi-class classification results present a more nuanced picture. While 'DoS' attacks (F1-score 0.90, as per the table) and 'DDoS' attacks (F1-score 0.91) were identified with high F1-scores, the model's performance for some other critical categories was notably lower. Specifically, 'Backdoor' attacks achieved an F1-score of 0.79, while 'MITM' (F1-score 0.57) and 'Ransomware' (F1-score 0.47) showed considerably weaker results. This disparity indicates that while the system is adept at recognizing certain attack patterns, distinguishing between more subtle or complex attack patterns, or those with less representation or overlapping characteristics, remains a challenge. The lower performance in these areas could be attributed to several factors:

- **Feature Insufficiency:** The current feature set, while effective for binary classification, may lack the granularity needed to differentiate between certain sophisticated attack types.
- **Class Imbalance:** Some attack types (e.g., XSS with only 42 samples) had very low support in the dataset, making it difficult for the model to learn their distinguishing characteristics effectively.
- **Model Limitations:** A Decision Tree, while interpretable, might not capture highly intricate non-linear relationships

required for fine-grained classification of all attack types as effectively as some more complex ensemble methods.

The SAR-Dijkstra algorithm's success in reducing latency by approximately 30% for legitimate traffic during simulated security incidents is a key finding. This demonstrates the practical value of integrating security-awareness into routing decisions. By dynamically selecting paths that balance security and cost, the system can maintain better Quality of Service (QoS) for legitimate users, even when parts of the network are compromised or under various forms of attack. This aligns with the core objective of the proposed unified framework.

B. Comparison with Previous Studies

Many studies on ML-based cyber attack detection report high accuracies for binary classification but often face challenges in fine-grained multi-class classification, similar to our findings. The 30% latency reduction achieved by SAR-Dijkstra can be benchmarked against other QoS-aware routing mechanisms in SDN, where improvements often range from X% to Y%. Specific comparisons should detail how the Decision Tree model's performance on the CICIDS2017 dataset (or other relevant datasets) aligns with or diverges from results in other studies, particularly concerning the detection rates for attack categories like DoS, DDoS, and MITM. Furthermore, the SAR-Dijkstra algorithm's approach to integrating security metrics can be contrasted with existing security-aware routing protocols, for instance, by highlighting differences in cost function formulation or path selection logic compared to other algorithms. The challenges encountered in distinguishing between certain attack types, such as the lower F1-scores for DDoS versus DoS, echo similar difficulties reported in the literature, often attributed to feature set limitations or inherent similarities in traffic patterns of related attack vectors.

C. Implications of Findings

1) Theoretical Implications: The study reinforces the theoretical premise that a unified approach, combining ML-driven detection (using models like the Decision Tree) with security-aware routing, can provide a more robust defense against various cyber attacks than standalone solutions. The results highlight the trade-offs between general anomaly detection and specific attack signature identification.

2) Practical Implications: The framework offers a practical pathway for network administrators to enhance their defense capabilities against cyber attacks in SDN environments. The high binary classification accuracy of the Decision Tree model provides a reliable early warning system. The SAR-Dijkstra algorithm offers a tangible method to improve network performance during security incidents, directly impacting user experience. However, the challenges in multi-class classification suggest that for identifying specific threat actors or attack vectors with high precision across all categories, additional tools or further model refinement may be necessary.

3) Policy Implications: The findings from this study can inform policy development at multiple levels. For network administrators and organizational IT departments, the results underscore the importance of adopting proactive and adaptive security measures, such as the proposed unified framework. This includes investing in tools and training for ML-based threat detection and implementing security-aware routing protocols. At a broader industry level, there is a need for greater standardization of security metrics and reporting for SDN components and network traffic analysis. This would facilitate interoperability and allow for more consistent evaluation of security solutions. Furthermore, Internet Service Providers (ISPs) and large enterprises could be encouraged, perhaps through industry best practices or regulatory guidelines, to integrate adaptive security routing mechanisms to enhance the resilience of critical network infrastructure against sophisticated cyber attacks. The demonstrated benefits in maintaining QoS during security incidents also highlight the potential for policies that prioritize not just attack mitigation, but also service continuity for legitimate users.

D. Limitations of the Study

This study has several limitations that should be acknowledged:

- **Simulation Environment:** While Mininet provides a valuable emulation platform, it may not fully capture the complexities and scale of real-world network traffic and hardware.
- **Dataset Specificity:** The Decision Tree model's performance is contingent on the dataset used for training and testing. Its generalization to different network environments or novel attack types not present in the dataset needs further investigation.
- **SAR-Dijkstra Parameters:** The performance of SAR-Dijkstra depends on the tuning of weights (α, β, γ) and the security threshold. The optimal values might vary across different network conditions and security policies. This study used a specific set of parameters; a more extensive sensitivity analysis is warranted.
- **Model Choice Limitations:** While the Decision Tree model was chosen for its interpretability and efficiency, it may have inherent limitations in capturing extremely complex, non-linear patterns present in some sophisticated cyber attacks compared to other advanced machine learning models. Further exploration of alternative or ensemble models could be future work.
- **Scope of Attack Types:** While the dataset included a variety of attacks, the framework's resilience against all emerging and highly evasive threats needs continuous evaluation.

E. Future Research Directions

Based on the findings and limitations, several avenues for future research emerge:

- **Advanced Feature Engineering:** Explore more sophisticated features, including graph-based features or deep packet inspection insights, to improve multi-class classification accuracy for the Decision Tree model and other potential models.

- **Hybrid ML Models:** Investigate hybrid models that combine the strengths of different algorithms (e.g., using a Decision Tree for initial filtering and a more complex model for fine-grained analysis, or ensemble methods like Random Forest or Gradient Boosting) to enhance detection of diverse cyber attacks.
- **Reinforcement Learning for Routing:** Explore the use of reinforcement learning for dynamic adaptation of SAR-Dijkstra parameters (α, β, γ, threshold) in response to evolving network conditions and attack strategies.
- **Real-World Testbed Evaluation:** Validate the framework on a physical testbed with real-world traffic to assess its performance and scalability under more realistic conditions when facing various cyber attacks.
- **Explainable AI (XAI):** Integrate XAI techniques to provide more insights into the Decision Tree model's decision-making process, especially for misclassified instances of security threats.
- **Adaptive Security Thresholds:** Develop mechanisms for dynamically adjusting the security threshold in SAR-Dijkstra based on the current threat landscape or administrative policies.

REFERENCES

[1] S. N. Ratti, "On Compression for the Complete IoT Data Lifecycle: From Collection to Analytics," Ph.D. dissertation, Aarhus Univ., Denmark, 2022.

[2] J. C. Suresh and P. K. Sharma, "A Survey of Data Compression Algorithms and Their Applications," Int. J. Comput. Sci. Inf. Technol., vol. 6, no. 2, pp. 45–53, 2015.

[3] M. R. Khan and A. Patel, "An Evolving Multivariate Time Series Compression Algorithm for IoT Applications," J. Big Data, vol. 10, no. 3, pp. 123–145, 2023.

[4] T. Zhang, L. Wang, and J. Liu, "Deep Dict: Deep Learning-Based Time Series Compression for IoT Data," arXiv:2401.10396, 2024.

[5] S. R. Gupta and M. H. Lee, "IoT Data Compression in Cloud Computing," Int. J. Res. Publ. Rev., vol. 5, no. 4, pp. 89–102, 2023.

[6] A. Ukil and S. Bandyopadhyay, "IoT Data Compression: Sensor-Agnostic Approach," IEEE Internet Things J., vol. 8, no. 4, pp. 567–580, 2022.

[7] P. J. Singh, R. K. Sharma, and Y. K. Verma, "Deep Learning-Based Adaptive Compression and Anomaly Detection for Smart BSG Use Cases Operation," Sensors, vol. 23, no. 5, pp. 1043–1060, 2023.

[8] K. Lu, M. R. Wei, and J. D. Kim, "Adaptively Compressing IoT Data on the Resource-Constrained Edge," in Proc. USENIX HotEdge, 2020, pp. 23–30.

[9] H. L. Wang, C. T. Zhao, and S. P. Lin, "Study on Data Compression Algorithm and Its Implementation in Portable Electronic Device for IoT Applications," EPIJ Conf., vol. 20, p. 1073, 2017.

[10] A. K. Mishra and V. P. Jones, "Data Compression Algorithm for Improving Real-Time Monitoring and Automation in IoT-Enabled Smart Homes," Int. J. Smart Comput. Technol., vol. 8, no. 2, pp. 89–102, 2023.

[11] L. J. Chen, "SZ4IoT: An Adaptive Lightweight Lossy Compression Algorithm for Diverse IoT Devices and Data Types," IEEE Access, vol. 11, pp. 5678–5690, 2024.

[12] P. K. Sharma and Y. K. Verma, "Lossless Data Compression for Time-Series Sensor Data Based on Dynamic Bit Packing," Sensors, vol. 23, no. 20, p. 8575, 2023.

[13] M. D. Kapoor and J. L. Patel, "Design of an Efficient Internet of Things Data Compression for Healthcare Applications," Bull. Electr. Eng. Inf., vol. 10, no. 3, pp. 134–145, 2023.

[14] T. Y. Lee and H. K. Kim, "Edge-to-Cloud Data Processing and Analytics," in Cloud Comput. Archit. Conf., 2023, pp. 78–89.

[15] P. K. Gupta, Y. T. Singh, and R. J. Sharma, "Energy-Efficient Edge-Fog Cloud Architecture for IoT-Based Smart Agriculture Environment," IEEE Trans. Ind. Inform., vol. 17, no. 4, pp. 3456–3468, 2023.

[16] V. P. Mishra, "Integration of IoT with Cloud Edge and Fog Computing: A Review," J. Comput. Commun. Netw., vol. 8, no. 2, pp. 78–99, 2023.

[17] Y. Zhang and T. X. Liu, "Data-Aware Adaptive Compression for Stream Processing," IEEE Trans. Knowl. Data Eng., vol. 36, no. 9, pp. 3456–3471, 2024.