Global Research journal of Natural Science and Technology (Grjnst)
Volume: 03 - Issue 4 (2025)
p.ISSN: 2790-7643 e.ISSN: 2790-7651

Intelligent SDN-Driven Multi-Layer Deep Learning Framework for Real-Time DDoS Mitigation and Secure Flow Management in Financial Networks

Khawaja Tahir Mehmood*
Department of Electrical Engineering, Bahauddin Zakariya University, Multan, 66000, Pakistan
ktahir@bzu.edu.pk
Raza Iqbal
Bahauddin Zakariya University, Multan, 66000, Pakistan
ali.raza@bzu.edu.pk

Crossponding Author: Khawaja Tahir Mehmood

DOI: https://doi.org/10.53762/grjnst.03.04.32.

Abstract

Today, digital financial infrastructures are facing critical security risks due to emergence of covert threat activities in the form of Distributed Denial of Services (DDoS) attacks by using adoptive Botnet infrastructures (network of hijacked devices that perform intense cyber-attacks on network clouds using adaptive application to continuously change the behavior as switch, router, IPS, and modifying attacks patterns to evade detection and have attack sustainability). The latest and intense DDoS attack toolkits for targeting Application Programming Interface (APIs), cloud services, and modern web applications by exploiting Hyper Text Transfer Protocol (HTTP-2) weakness are DDoSia and MantisBot. These DDoS attacks have volumetric, multi-vector, and low-rate features to target banking APIs working on the 7th layer, which traditional machine learning algorithms cannot mitigate. This research proposes a multi-layer Software Define Networking (SDN)-Pox controller based structure installed on Mininet featuring an intelligent real time a hybrid deep learning framework for

DDoS attacks mitigation based on three algorithms and Honeypot implementation: (1)Long Short Term Memory (LSTM)-Auto encoder (LSTM is a recurrent neural network performing sequential data analysis and Auto encoder reconstruct the input and flag anomalous flow when reconstruction error is high) and next algorithm in proposed framework is (2) The 25-featured based Optimized XG-Boost (XG-Boost 25) Classifier (gradient boost decision tree algorithm deploying 25 features to detect the flow flagged by LSTM-Auto encoder is benign or malicious) and last step in multi-layer framework is (3) Graphical Neural Network (GNN) algorithm (capture spatial relationship between hosts, switches and flows for detecting complex attack patterns having lateral coordinated movement across nodes). Upon detection of DDoS attacks, the flow is not just rejected; to have detailed attacker deception and forensic logging, the malicious flow is forwarded to an isolated Honeypot environment. With the proposed multi-layer framework, the accuracy of 98.9%, 160ms of detecting time,30ms of mitigation latency, 1.7% of packet loss efficiency and minimum value of J-index (explaining the overall degradation and risk to system performance under DDoS attacks combining multiple matrices to single scalar value) is achieved as compared to traditional mitigation framework based on Random Forest, signature-based Intrusion Detection System (IDS), etc.

Keywords:

Software Define Networking (SDN); LSTM-Auto encoder; XG-Boost 25; Graphical Neural Network (GNN); DDoS Attacks Mitigations; Digital Financial Infrastructure.

1 Introduction

The advanced Distributed Denial-of-Service (DDoS) attacks with extreme levels of complexity and scaled as undetectable by traditional mitigation techniques are becoming a greater menace

to the financial industry. The greatest DDoS assault ever recorded occurred in 2022 when the Mantis-Botnet launched a Hyper Text Transfer Protocol (HTTPs) flood that peaked at 26 million Requests per Second (RPS) [1]. Thousands of attacks were launched across industries by the Mantis-Botnet alone, with the finance industry being one of the most heavily targeted [2]. In 2023, attackers used an HTTP/2 zero-day (CVE-2023-44487, known as "Rapid Reset") to create hyper-volumetric DDoS floods that reached a high of 201 million (RPS), which was almost eight times the previous record [3]. Later, in the year 2023 the Google reported one of these new HTTP/2-based attacks that approached 398 million (RPS), the greatest application-layer DDoS attack ever documented. These attacks were much greater than any previous Layer-7 attacks [4]. Additionally, threat actors are using crowdsourced platforms and botnets to launch politically motivated assaults. For instance, waves of DDoS attacks on European banks and government websites have been planned by the pro-Russian group NoName057(16), even employing a crowdsourcing attack tool called DDoSIA (now at version 3.5) [5-6]. Modern DDoS threats against banking services are not only huge but also multi-vector and sneaky, frequently imitating legal traffic to avoid detection. This is demonstrated by hacktivist DDoS barrages (such as DDoSIA and Mantis) and new HTTP/2 abuses [7]. Under volumetric attacks, traditional firewalls can become sites of failure because they keep track of connections' states, making them vulnerable to state-table depletion. Once their connection tables are filled with millions of malicious flows, they crash [8]. Firewalls and outdated IPS frequently use static rules and fixed thresholds, blocking ports or IPs if a limit is exceeded, even when they are not completely failing. Attackers can readily take advantage of this direct technique by sending messages that are just below the threshold or by changing patterns. They can even block legitimate users, resulting in self-inflicted outages [9]. Today, the need of the hour is to have a

multi-tier machine learning based mitigation infrastructure having features to detect and mitigate DDoS threats, with features of being multi-vector and sneaky, frequently imitating legal traffic to avoid detection. The research study [10] proposes a model that continually scans aggregated traffic time-series for early indications of anomaly using unsupervised deep learning (LSTM). Significant reconstruction errors signify departures from the regular behaviour of long short-term memory networks, which learn and reconstruct the normal temporal patterns in measurements (e.g., per-port packet rates). At the network level, this module offers a Zero Trust-aligned [11]"always verify" mechanism that generates alarms if traffic patterns deviate from the baseline, even for attack types that haven't been detected before [12]. The research study [13] provides a method gradient-boosted decision trees "XGBoost-25, providing quick inference and excellent accuracy. The XG-Boost model uses 25 criteria to categorize flows as either benign or a particular kind of attack (such as an HTTP flood or SYN flood). This uses a model trained on labelled attack data to produce a high-confidence decision on malicious flows. The research study [14] provides a literature review for an analytical module that uses a Graph Neural Network (GNN) to correlate alarms and identify patterns at the botnet scale. The GNN module improves the detection of widespread attacks by classifying suspicious flows and determining whether they are part of a common botnet or campaign [15]. From per-flow detection to identifying dispersed assault campaigns, it successfully provides a group context, which is an essential skill against botnets like Mantis or DDoSIA-driven crowd [16]. In the research article [17-18], the mitigation strategy is discussed in the form of a Honeypot segment that is a programmable active defence module that uses SDN to reroute attack traffic into separate honeypots and initiate deception reactions. The controller uses OpenFlow to dynamically rewrite flow rules in order to divert hostile flows from production

servers to a hardened honeypot network once an attack has been verified. By imitating banking services, these honeypot servers engage and innocuously absorb the attackers' traffic. In the research study [19-20], prioritizes real-time policy adaptation and fine-grained monitoring, which align with Zero Trust's need for ongoing risk assessment and policy enforcement.

1.1 Research Objectives

The motivation of this research is to mitigate the botnet-oriented DDoS attacks that are imposing severe cyber-attacks on digital financial applications and eroding the trust of Artificial Intelligence (AI) organizations. However, the number of objectives to be achieved in this research is as follows:

a) To model an SDN-based security framework that provides dynamic DDoS mitigation policies in cloud-hosted financial applications.
b) To design an algorithm for the SDN-Pox controller having features of a multi-layer hybrid framework (LSTM-AE, XG Boost-25, GNN).
c) To mitigate Botnet-oriented DDoS attacks, LSTM-AE should be designed to detect the deviations in flow-level behaviors using a time-series network. The implementation of XGBoost-25 classifier for supervised verification of anomalous traffic flows and (GNN) to detect coordinated botnet-driven attack patterns.
d) To investigate malicious traffic patterns without service disruption, the Honeypot-based mitigation should be implemented by the SDN-Pox controller.
e) To judge the performance and effectiveness of the proposed framework against Botnet-based DDoS attacks, the comparative analysis is performed with traditional detection frameworks (e.g., static rule-based IDS, Random Forest classifiers).

f) To determine the J-index under real-time Botnet-based DDoS attack for the SDN mitigation network with the proposed hybrid multi-layer algorithm, and compare this factor (J) with traditional detection methods.

1.2 Work Contribution

In this research, the Botnet (DDoSia and MantisBot) oriented DDoS attacks on digital financial infrastructures are mitigated by design of a multi-layer Software Defined Networking (SDN)-Pox controller-based structure featuring an intelligent, real-time a hybrid deep learning framework with Honeypot implementation. The detection process in the proposed framework is performed by implementing three hybrid deep learning algorithms in sequential order that are (1) Long Short-Term Memory Autoencoder (LSTM-AE), (2) XG-Boost-25, and (3) Graphical Neural Network (GNN) algorithm. In the (LSTM-AE) algorithm, the (LSTM) is a recurrent neural network performing sequential data analysis, and Auto autoencoder reconstructs the input and flags anomalous flow when reconstruction error is high. While the 25-feature-based Optimized XG-Boost (XG-Boost 25) Classifier is the gradient boost decision tree algorithm deploying 25 features to detect the flow flagged as benign or malicious, and (GNN) algorithm captures spatial relationships between hosts, switches, and flows for detecting complex attack patterns having lateral coordinated movement across nodes. All above mentioned algorithms are designed to operate in sequential order to perform Honeypot mitigation. If flow is detected as malicious by LSTM-AE, then it is forwarded to XG-Boost-25 for further classification of malicious flow as DDoS or normal, and after this, it is forwarded to the GNN algorithm process to detect whether the DDoS attack was by

an isolated node or by a Botnet mechanism. Upon detection of DDoS attacks (by isolated node or Botnet mechanism) with the proposed hybrid multi-layer framework, the mitigation is performed; not just by rejection of the flow, but to have detailed attacker deception and forensic logging, the malicious flow is forwarded to an isolated Honeypot environment. The block diagram of the proposed method framework is shown in Fig.1.The effectiveness of the proposed framework is evaluated by the calculation of parameter (J), as explained in Eq.A. The function (J) explains the overall degradation and risk to system performance under DDoS attacks, combining multiple matrices into a single scalar value. The proposed algorithm will perform effectively when (J) is minimum, and if the value of the function (J) is increased, the system will be at risk and will be slow to detect, ineffective at mitigating, lose bandwidth with greater chances of dropping packets, a greater number of connections Failure, and the longer response times.

$$F(J_{min}) = \alpha(1\text{-}A) + \beta L_d + \gamma L_m + \delta \left(\frac{P_{loss}}{Bd_{ava}}\right) + \varepsilon T_{API\_response\ time} + \zeta T_{connection\_failure} \quad (A)$$

Where:

A = Accuracy (should be maximum), and factor (1−A) becomes small as accuracy improves

L_d & L_m = Detection and mitigation time (these terms should be minimal)

P_{Loss}/Bd_{ava} Packet loss per unit retained bandwidth (this term should have a maximum value)

$T_{API\_response\ time}$ = API response time (this term should be minimum)

$T_{connection\_failure}$ = TCP connection failures (this term should be minimum)

(α,β,γ,δ,ε,ζ) = weights of Eq. A and their sum should be equal to 1

1.3 Paper arrangement:

In section 2, the different techniques to manage the Botnet-oriented DDoS attacks are reviewed to obtain the research gaps, and the proposed hybrid multi-layer

framework based on SDN infrastructure is contrasted with traditional DDoS attack mitigation methods. Section 3 presents the algorithm and methodological procedures used to achieve the aforementioned objectives. Section 4 presents the outcomes of the system with the proposed framework under Botnet-based DDoS attacks, and Section 5 presents the conclusion.

2 Literature Review

Modern digital financial infrastructures are now very much security compromised due to sophisticated DDoS attack frameworks like DDoSia and MantisBot having liberties in attacking cloud-hosted services, RESTful APIs, and contemporary online apps. The patterns of their DDoS attacks include low-rate, multi-vector, and high-volume features that are especially successful against Layer 7 banking APIs. The traditional machine learning-based defence mechanisms are ineffective at identifying and preventing such complex, protocol-aware threats, because they are unable to adjust to the changing temporal and structural complexity of these attacks. Today, the number of research studies and frameworks is being tested in testing conditions to make the digital financial application more secure and effective against these modern DDoS attacks. In this section, the findings of a number of pertinent recent research methods are highlighted in Tab.1, which also compares them to a proposed hybrid multi-layer framework.

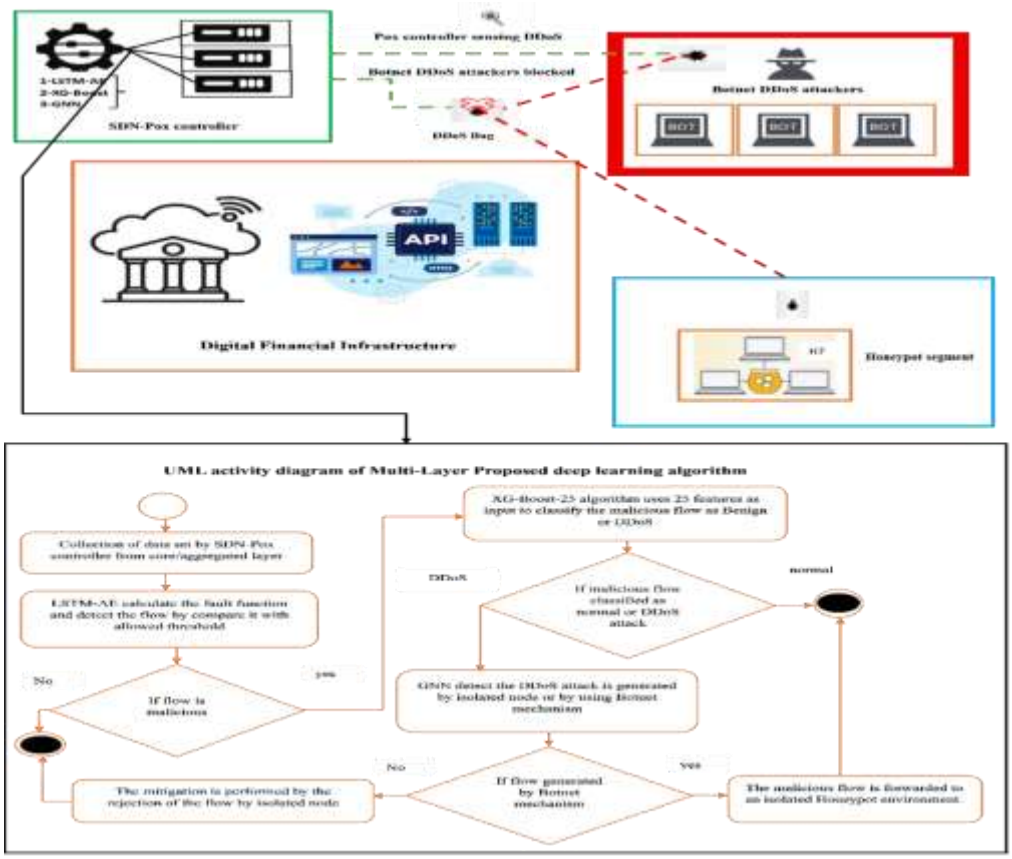

Figure 1: Block diagram of the proposed framework for encrypted virtual UE data storage

TABLE 1: Comparison between traditional DDoS attack mitigation techniques with the proposed hybrid multi-layer framework.

Authors	Contributions	Limitations and Comparison with Proposed Framework
L.Ming.et.al.[21]	This research focuses on DDoS attack types and security risks related to the HTTP/2 protocol and proposes potential areas of research for DDoS attack detection.	Survey/analysis of HTTP/2 and DDoS risks offers a taxonomy but no real-time control or Layer-7 correlation. Compared to this, the Proposed Hybrid ML-SDN Method achieves online detection (~160 ms) and mitigation (~30 ms) with LSTM-AE + XGBoost-25 + GNN, and enforces per-flow OpenFlow policies and honeypot redirection—capabilities absent in the study.
S. Salmi.et.al.[22]	This research proposes a machine learning-based method for identifying and preventing DDoS attacks at the	Application-layer ML relies on fixed thresholds/features and lacks controller-wide visibility; resilience to low-rate, mimicry traffic is limited. Our method couples unsupervised LSTM-AE temporal scoring with supervised XGBoost-25 and SDN enforcement, cutting

	application layer, which are an important risk to web-based services.	detection latency and reducing FPR (≤1.1%) while preserving ≥810 Mbps under attack.
N. Kathirkamanathan.et.al [23]	The study suggests employing deep learning models with stacked Long Short-Term Memory (LSTM) to counteract DDoS attacks that target transactions in financial services.	Stacked LSTM detects anomalous transactions but omits cross-host correlation and network-level mitigation. We extend LSTM with XGBoost-25 for precise flow labelling and GNN for botnet grouping, then apply POX flow rules (drop/redirect) to contain sources, delivering Zero-Trust micro-segmentation missing in the baseline.
E. Leka et.al.[24]	In order to identify and stop botnet-based DDoS attacks on digital financial infrastructures, the study suggests a novel Web Application Firewall that utilizes the use of blockchain-based algorithms and neural networks.	Blockchain-WAF approach adds overhead and typically operates out-of-band; limited SDN orchestration and no deception loop. Our SDN-resident pipeline operates in-band, installs rules in switches within tens of ms, and diverts attacks to high-interaction honeypots, yielding lower packet loss (≈1.7%) and faster service recovery.
S.Paliwal.et.al.[25]	This research topic explores machine learning techniques developed to identify and stop DDoS attacks.	Generic ML review lacks deployable controller logic, temporal modelling, and botnet-scale attribution. Proposed Hybrid ML-SDN provides an executable stack (POX) with LSTM-AE (temporal), XGBoost-25 (flow-level), GNN (graph-level), and OpenFlow mitigation—demonstrably higher accuracy (98.9%) and sub-second actuation.
K. Sivakumar et.al.[26]	This study looks at how application-layer DDoS assaults that get around rate restrictions can result from complex multi-client attacks on high-workload API endpoints.	Focus on rate-limit evasion at busy APIs without coordinated multi-client inference; mitigation is largely static. Our graph-aware GNN links dispersed low-rate sources, enabling coordinated quarantine and honeypot redirection; bandwidth retention improves (≥810 Mbps) versus static rate-limit defenses.
S. Dasari. et.al.[27]	In order to accurately characterize DDoS attacks in a distributed network, the study offers a hierarchical machine learning technique with hyperparameter optimization.	Hierarchical ML improves labelling but depends on tuned hyperparameters and lacks Zero-Trust enforcement and deception. We fuse anomaly→classification→correlation with SDN micro-policies and honeypots, yielding faster TTM (≤30 ms to mitigate) and reduced collateral blocking compared to coarse IP/port bans.
U. Islam.et.al.[28]		SVM on financial traffic is sensitive to feature drift and class imbalance; limited Layer-7

	The study proposes implementing machine learning models, particularly SVM, to detect DDoS attacks on financial institutions.	coverage and no SDN hooks. Our ensemble (LSTM-AE + XGBoost-25) is more robust to drift, augmented by GNN correlation; POX installs per-flow rules—closing the detect-to-act gap absent in SVM-only designs.
M.V. Prashanth et.al.[29]	The study examines current research on deep learning and machine learning methods for detecting and stopping distributed denial of service (DDoS) attacks, which constitute serious risks to digital financial systems.	Survey of DL/ML lacks practical SDN integration and deception-based response; no metrics on latency/availability. We report concrete metrics (98.9% accuracy, ~160 ms detection, ~30 ms mitigation, 1.7% loss) and realize inline SDN enforcement with honeypot telemetry to continuously harden models.
E.M. Manaa. et.al.[30]	In order to detect and mitigate DDoS attacks in IoT environments, the study proposes using machine learning and deep learning methods.	IoT-centric DL pipelines often assume homogeneous endpoints and do not address financial Layer-7 semantics; mitigation unspecified. Our controller-resident pipeline targets banking APIs, correlates multi-vector floods via GNN, and enforces OpenFlow policies with honeypot diversion—improving API latency recovery (<290 ms) under attack.
P.S. Saini.et.al.[31]	The study suggests using machine learning to identify and categorize recent developments like SID DoS and HTTP flood DDoS attacks.	Binary ML for new attack classes (e.g., SID DoS, HTTP floods) can overfit and lack graph context; response is detector-only. We complement supervised XGBoost-25 with unsupervised LSTM-AE and graph-level GNN, then activate POX policies—lowering FPR and preventing state-table exhaustion seen in appliance-based mitigations.
M.Aslam.et.al.[32]	In order to identify and prevent DDoS attacks in SDN-enabled IoT networks, the study suggests an adaptive machine learning-based system.	Adaptive ML for SDN-IoT improves detection but overlooks deception and Zero-Trust micro-segmentation; enforcement granularity is limited. Our method adds honeypot-aided containment and per-flow segmentation, yielding higher availability and reduced collateral impact.
Djenna.et.al.[33]	Because traditional security methods are insufficient, the article suggests an AI-based technique for recognizing and preventing DDoS attacks on IoT networks.	AI proposal for IoT DDoS lacks controller timing guarantees and botnet attribution; mitigation unspecified. We provide end-to-end timings (≤160 ms detect, ≤30 ms mitigate), botnet clustering via GNN, and concrete OpenFlow rules for drop/redirect—closing the loop from analytics to action.
O. Aslan. et.al.[34]	The paper offers a thorough analysis of cybersecurity risks, vulnerabilities, attacks, and defences. It also highlights the necessity of creative	Broad cybersecurity survey; no specialized, low-latency SDN pipeline or deception workflow. Our integrated controller pipeline operationalizes the survey's recommendations

	ways to identify complex attacks like DDoS.	with measurable gains in accuracy, latency, and resiliency.
T.E. Ali.et.al.[35]	To be able to identify DDoS attacks in software-defined networks, this article examines machine learning computations.	ML for SDN DDoS focuses on classifiers but omits the temporal anomaly front-end, graph correlation, and deception; actions are often offline. We chain LSTM-AE→XGBoost-25→GNN with immediate POX rules, enabling real-time Zero-Trust enforcement absent in classifier-only work.
J.G.A. Rivera. et.al.[36]	In order to identify transport and application layer DDoS assaults against IoT devices, the article suggests machine learning and deep learning models.	IoT transport/app-layer ML improves per-device detection but misses coordinated campaigns and lacks SDN-based response. GNN clusters coordinated sources targeting h6; POX enforces per-source policies and honeypot redirection to preserve core capacity.
V.S. Tharunika.et.al.[37]	This research proposes a technique for identifying and thwarting botnet-based DDoS attacks in IoT networks that makes use of machine learning and deep learning.	Botnet-oriented ML/DL identifies patterns but typically out-of-band; mitigation is not coupled to the network fabric. Our in-band SDN approach ties GNN detections to OpenFlow actions in milliseconds, reducing attack dwell time and blast radius.
J. Singh. et.al.[38]	This study examines DDoS mitigation and detection techniques in SDN networks, emphasizing unresolved problems and difficulties.	SDN DDoS surveys highlight gaps (false positives, controller bottlenecks) but provide no deployable Zero-Trust design. We address these by cascading detectors, throttling at edges, and isolating malicious flows—maintaining ≥810 Mbps throughput under load.
N. Mishra. et.al.[39]	With an emphasis on reducing the risk of DDoS assaults, this paper offers a thorough analysis of intrusion detection methods, security issues, and attacks for the Internet of Things.	IoT IDS reviews emphasize signatures/rules with limited generalization to novel L7 vectors and HTTP/2 abuses. Our LSTM-AE captures unseen temporal deviations; XGBoost-25 learns modern feature importances; GNN adds campaign-level context; together they outperform signature-centric baselines.
H. Hartono.et.al.[40]	In order to identify and eliminate sophisticated persistent threats in cloud computing infrastructure, this article suggests a multi-layered AI-based approach.	Multi-layer AI for cloud APT lacks SDN enforcement for DDoS and does not include deception. We embed the stack in POX for per-flow control and add honeypot traps, achieving faster containment and richer threat intel.

T. Yerriswamy et.al.[41]	An effective hybrid protocol framework for detecting and stopping DDoS assaults on web applications is proposed in this study.	The hybrid protocol framework enhances web-app detection but lacks graph correlation and SDN micro-segmentation; mitigation is typically coarse. Our method's GNN + SDN rules realize coordinated, low-collateral containment with documented latency gains.
R. Redekar. et.al.[42]	In order to address the serious security threats that digital financial infrastructures face, the article examines DDoS attack detection methods, difficulties, and contemporary practices.	Financial DDoS practices review identifies challenges, but no real-time controller implementation or deception strategy. Proposed Hybrid ML-SDN delivers a controller-resident, real-time pipeline with honeypot diversion and measured improvements in accuracy/latency over legacy stacks.
N. Myungaicela-Naula et al. [43]	Transport and application layer DDoS attacks can be efficiently detected via an SDN-based architecture that makes use of machine learning and deep learning models.	SDN-based ML/DL shows promise but often evaluates in narrow scenarios without deception or Zero-Trust policies. Our framework broadens scope to multi-vector L3–L7, adds GNN correlation and honeypot mitigation, and quantifies benefits (~160 ms detection; ~30 ms mitigation; 1.7% loss).

2.1 Research Gaps:

The following research gaps are identified in correlation to the research objectives mentioned in section 1.1, taking into account the contribution of the most recent research methodologies and comparing them with the proposed hybrid multi-layer framework in Table 1:

1. The traditional machine learning algorithms are no longer potent in providing security to the digital financial application from covert cyber attacks featuring volumetric, multi-vector, and low-rate activities to target banking APIs working on the 7th layer.
2. The hybrid multi-layer deep learning algorithm should be implemented on the SDN controller to mitigate the DDoS attacks generated by Botnet-oriented tools (e.g., DDoSia and MantisBot). The first layer of the algorithm should be able to detect the malicious flow by monitoring the logs of core layer switches. The second layer should be able to classify the suspicious flow as (1 as DDoS and 0 as Benign). The third layer of the algorithm should be able to determine the DDoS source (attack launched by an

isolated node or a Botnet mechanism is utilized)

3. The flow is not to be just rejected but directed to the Honeypot segment. The Honeypot can mimic the real banking platform, believing the attack that they are in a real banking server, and keeping them engaged, and perform forensic analysis of their attack and forward all the relevant details to the SDN controller so that such a type of Botnet DDoS attack can easily be detected and immediately rejected without undergoing a multi-layer operational process.

The proposed algorithm to mitigate the Botnet-oriented DDoS attacks is designed to correlate the research gaps and research objectives mentioned in the above section of this research work.

3 Methodology of the Proposed Framework for Encrypted Virtual Data Storage

This section explains the working of the proposed multi-tier framework based on SDN-Pox controller using a hybrid deep learning approach for real-time DDoS attack mitigation. The detail is as follows:

3.1 Topological Structure of the Proposed Model:

In this research article, the proposed multi-layer SDN-based framework is built on Mininet, controlled via Pox controller featuring a hybrid deep learning (LSTM-Autoencoder, XG-Boost 25, and GNN) detection and mitigation pipeline. The topology of the proposed framework features five layers, namely (1) client zone layer, (2) edge layer, (3) core layer, (4) application layer, and (5) security layer, and one SDN-control plane. The topological arrangement to mimic the whole scenario, starting from client access to banking API, is shown in Fig.2. The client zone layer consists of hosts (h1-h5) that represent both real users (h4 and h5) and DDoS attackers(h1-h3) that initiate flow toward the banking App. The edge

layer consists of open flow-based edge switches (S4 and S5) to direct the flow of users (real and attackers) toward the API of the banking App. The core layer consists of an aggregation switch (S1) that combines all the flows from edge switches and enables centralized monitoring and redistribution based on SDN control instructions. The application layer consists of a server switch (S2) that handles the API requests for the banking application server (h6 acting as a centralized server). The security layer consists of a Honeypot segment switch (S3) that performs the redistribution of the suspicious flow and mirrors these flows for further investigations. The SDN-Pox controller plane uses the hybrid proposed framework to detect the DDoS attacks on the centralized server(h6) via continuous monitoring of the aggregation switch (S1) and mitigate the anomalous flow to the Honeypot segment.

3.1.1 Working of each Component in Proposed Framework Topology:

In the client zone layer, the hosts (h1-h3) are declared as DDoS attack generators on the centralized banking server(h6). At host (h1), to introduce the DDoS attack, the Synchronized (SYN) Flood Mode attack method is utilized. In this case, the connection table is flooded by sending a series of Transmission Control Protocol Synchronized (TCP-SYN) packets to the centralized server without completing the three-way handshake process, thus exhausting the resources, and this process leads to the generation of a DDoS scenario for real users. While at host (h2), the HTTP/2 attack mode is selected, in this method the thousands of lightweight requests (low bandwidth and high processing cost) are made that consume server resources and generate a DDoS scenario for real users. In the host (h3), the Slowloris attack method is utilized, which involves the process of slowly sending the incomplete header packets, forcing the server to keep the connection open with all its threads tied up and denying access to the

real users. The malicious flow from these three hosts (h1-h3) is sent to the server using the edge switch (S4). While the hosts (h4-h5) are real users who may access the APP for financial transactions using the open flow edge switch (S5). The core/aggregation switch (S1) provides the aggregation to all incoming flows from edge switches (S4 and S5). It is continuously queried by the SDN-Pox controller to have a global view of traffic flow that acts as a basic building block in the form of a data set for the proposed hybrid deep learning model to detect and mitigate the DDoS attack before it reaches the application layer (server h6). The switch (S2) is an application-layer switch that connects to the main server (h6). When the flow is declared benign by the proposed hybrid deep learning model, the application switch provides access to the server of the banking App to perform legitimate actions. When the flow is declared be malicious by the proposed hybrid deep learning model then this flow is forwarded to the Honeypot segments switch (S3) that forward it to the safe separate Honeypot segments (h7) where these flows are further analyzed to perform the forensic analysis of the attack so that this form of attack replicated in future, then it should be avoided at initial level (aggregation switch S1).

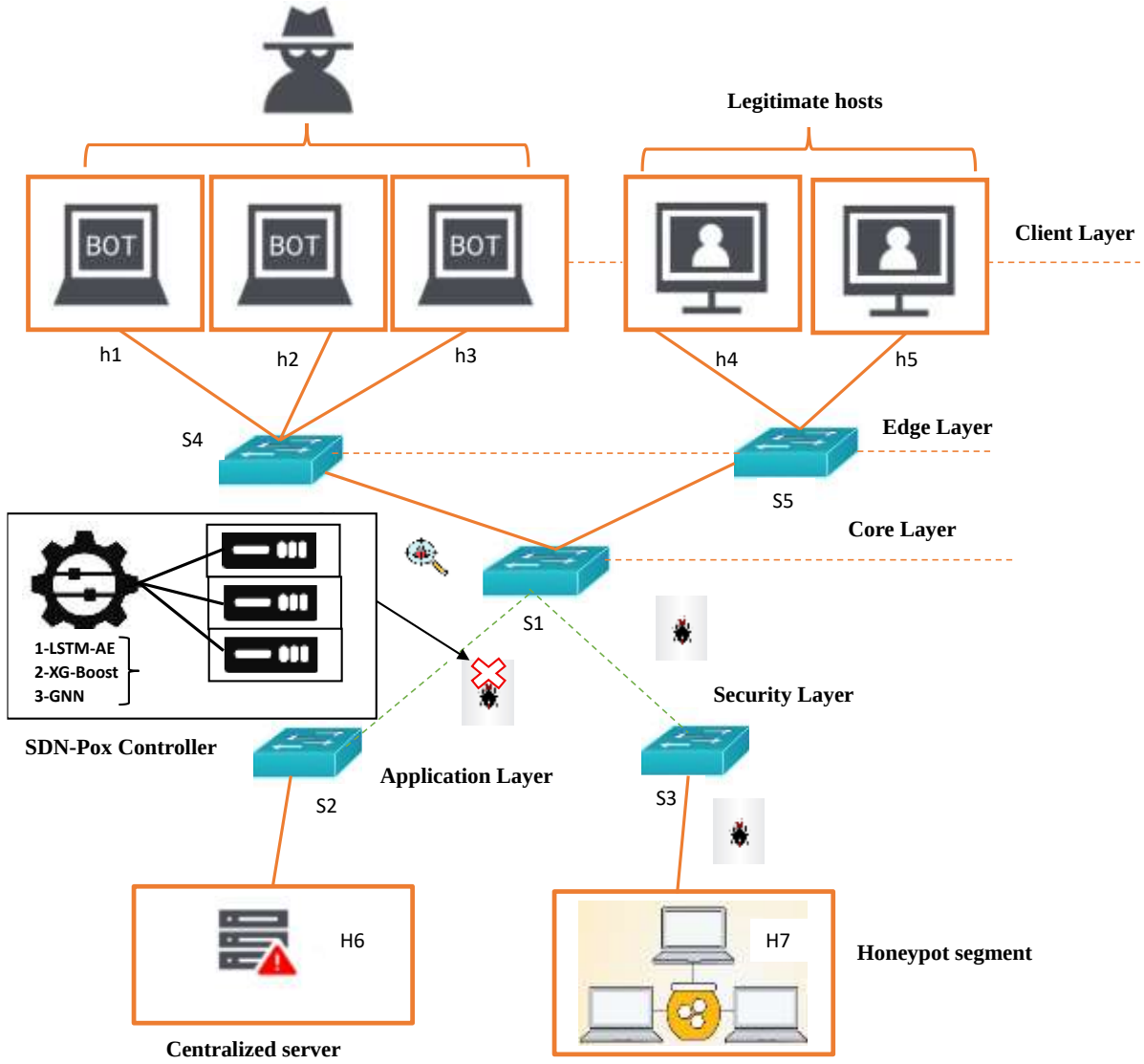

Figure 2: Topological arrangement of the proposed framework designed on Mininet.

3.2 Working of Proposed Hybrid Multi-Layer Framework:

This research proposes a multi-layer Software Define Networking (SDN)-Pox controller-based structure installed on Mininet featuring an intelligent, real-time a hybrid deep learning framework for DDoS attacks mitigation based on three algorithms and Honeypot implementation. The detection process in the proposed framework is performed by implementing three hybrid deep learning algorithms in sequential order that are (1) Long

Short-Term Memory (LSTM)-Autoencoder, (2) XG-Boost-25, and (3) GNN algorithm. If flow is detected as malicious by LSTM-AE, then it is forwarded to XG-Boost-25 for further classification of malicious flow as DDoS or normal, and after this, it is forwarded to the GNN algorithm process to detect whether the DDoS attack was by an isolated node or by a Botnet mechanism. Upon detection of DDoS attacks (by isolated node or Botnet mechanism) with the proposed hybrid multi-layer framework, the mitigation is performed; not just by rejection of the flow, but to have detailed attacker deception and forensic logging, the malicious flow is forwarded to an isolated Honeypot environment. The detailed working of each algorithm in the proposed framework is as follows:

3.2.1 Working of the First Deep Learning Algorithm in the Proposed Algorithm (LSTM-Autoencoder):

The LSTM-Autoencoder involves two further deep learning algorithms (1) Long Short-Term Memory (LSTM) is the recurrent neural network that detects the following abnormalities from previous time steps in sequence, namely:(a) sudden spikes in data flows, (b) repeated behavior, (c) delayed attacks. The Autoencoder is also a neural network method whose encoder compresses data and then reconstructs the decoder side and checks the reconstruction error. If the error is greater than the allowed threshold, then the flow is considered anomalous and forwarded to XG-Boost-25 for further verification. The detailed working of this algorithm in a combined form is explained in the form of the following steps:

Step#1-Preparation of Data Sets:

The SDN-Pox controller generates the flow sequence data set by continuously monitoring the data flow at the aggregation/core layer switch (S1). The flow sequence data sets are based on

the function shown in Eq.1

(Flow Sequence: $X\epsilon R^{T\times F}$): where *(T: number of time steps =10sec and F: features per step = 6)* (1)

The flow sequence matrix, illustrated in Eq.1, contains the normal traffic flow statistics observed for 10 seconds, and for each second, the 6 features are extracted as shown in Tab.2.

TABLE 2: Representing the 6 features extracted from the aggregation switch (S1) to generate the data set ($X\epsilon R^{10\times 6}$)

Sr.NO	Features per Step	Description
1	Packet_count	number of packets in the last T seconds
2	Byte_count	Total bytes transferred
3	Time_Avg (Inter-packet)	Average time between the packets
4	TCP flags _count	SYN (synchronized), RST (reset), FIN (finished) packets
5	Flow duration	Time since the first packet arrived
6	Entropy _IP	Variation in source IPs in the local group (if aggregated)

Step#2-Building of LSTM-Autoencoder:

This step involves four important features: (1) using two encoders for compression of temporal behavior, (2) using a repeat vector for reconstruction, (3) two decoder layers, and (4) a time distribution dense layer to reconstruct the original features. The Pseudocode to perform the above-mentioned task, as defined in four feature steps, is shown in Algorithm 1.

Algorithm 1: Pseudocode for the building of LSTM-Autoencoder

	ALGORITHM
1	# Calling library functions

```
From keras. models import Sequential
from keras. Layers import LSTM, Repeat Vector, Time Distributed, Dense.
2  # Initialize the Sequential function (Tensor Flow)
   model = Sequential ([
3  # First LSTM Layer (Encoder)
   LSTM (64, return_sequences=True, input_shape= (10, 6)),
4  # Second LSTM Layer (Encoder to Bottleneck)
    LSTM (32, return_sequences=False),
5  #  Repeat Vector (Bottleneck Expansion)
     Repeat Vector (10),
6  # Third LSTM Layer (Decoder)
   LSTM (32, return_sequences=True),
7  # Fourth LSTM Layer (Decoder)
    LSTM (64, return_sequences=True),
   # Time Distributed Dense Layer (Output)
8   Time Distributed (Dense (6))
          ])
9  #  Compile the model with optimizer and loss function
   model. Compile (optimizer='adam', loss='mse')
```

Comments:

This LSTM Autoencoder model performs anomaly detection by learning and reconstructing time-series sequences. Each input sequence is designed to have 10-time steps and 6 features each. These input sequences are processed and compressed into a lower-dimensional context vector through an encoder composed of two LSTM layers (64 and 32 units). For reconstruction of the sequence, this vector is then expanded back to the original time dimension using a repeat Vector layer and decoded through two LSTM layers (32 and 64 units). To have outputs sequence with the same shape as the input, a Time Distributed Dense layer function is used. Outputs the final sequence with the same shape as the input. The model is trained to minimize the mean squared error (MSE) between the input and output, enabling it to identify anomalies based on high reconstruction error.

Step#3-Training of LSTM-Autoencoder:

The LSTM-Autoencoder is trained on normal traffic, and the set of commands used for the training of the LSTM-Autoencoder is shown in Algorithm 2

Algorithm 2: Pseudocode for the training of LSTM-Autoencoder

	ALGORITHM
1	model. Fit (X_train_normal, X_train_normal,
2	epochs=50, batch_size=32, validation_split=0.1)

Comments:

This LSTM Autoencoder model is trained on normal sequential data steps with 10-time steps and 6 features each. The model goes through the entire data set 50 times to perform better learning. In order to provide fast processing, the data sets are divided into smaller chunks called batches, as in this case we have 32 sequences in each batch, and the batch shape is (32,10,6). The 10% of training data is used for testing of model to avoid the overfitting phenomenon.

After the training of the model on normal data, now next step is to compute the fault function (as shown in Eq.2), which is the mean square error, and the threshold θ_{AE} from the validation split function (in our model, the value =0.0075). The reconstruction error (ϵ)is shown in Eq.3.

$$Mean\ Squared\ Error\ (MSE) = \frac{1}{n}\sum_{i=1}^{n}(x_{i-}\ x_i^-)^2 \quad (2)$$

$$(\epsilon) = (MSE) > \theta_{AE},\ then\ the\ model\ detects\ anomalous\ flow \quad (3)$$

In Eq.2, X_i is the actual sample value and X_i^- is the estimated sample value. If the value of

reconstruction error (ϵ)is greater than the (θ_{AE}) limit, the flow is considered anomalous and forwarded to XG-Boost-25 for further detection.

3.2.2 Working of the second Deep Learning Algorithm Optimized XG-Boost (XG-Boost 25) Classifier in the Proposed Algorithm:

The Data set having (reconstruction error (ϵ) = Mean Square Error(MSE) > θAE) is considered as anomalous and is received from the LSTM-Autoencoder to XG -Boost algorithm that is integrated with three important features: (1) preprocessed into the 25-feature input vector (2) classification of flow in form of 1 or 0 (1as anomalous flow and 0 as normal flow) using 25 features as an input vector (3) mapped to real-time mitigation actions by the POX controller. The list of 25 features as an input to the XG-Boost neural network to classify the flow as DDoS or Benign is shown in Tab.3.

TABLE 3: The list of 25 features as an input to the XG-Boost neural network

Feature No.	Feature Name	Description
Feature#1	Flow_Duration	Time between first and last packet
Feature#2	Total Fwd_Packets	Number of packets in the forward direction
Feature#3	Total Backward_Packets	Number of packets in the reverse direction
Feature#4	Total Length of Fwd _Packets	Bytes sent forward
Feature#5	Total Length of Bwd _Packets	Bytes sent backward
Feature#6	Flow _Bytes/s	Overall byte rate
Feature#7	Flow _Packets/s	Packet rate
Feature#8	Fwd _Packet Length Mean	Avg forward packet size

Feature#9	Bwd _Packet Length Mean	Avg backward packet size
Feature#10	Flow _IAT Mean	Avg inter-arrival time
Feature#11	Flow _IAT Std	Standard deviation of IAT
Feature#12	Fwd _IAT Total	Total inter-arrival time (fwd)
Feature#13	Bwd _IAT Total	Same for backward
Feature#14	PSH_Flag Count	Packets with the Push flag
Feature#15	ACK_ Flag Count	Acknowledgment in flow
Feature#16	URG_Flag Count	Urgent (URG) packets
Feature#17	Init _Win Bytes (Fwd)	TCP initial window (fwd)
Feature#18	Init _Win Bytes (Bwd)	Same for backward
Feature#19	Avg Packet Size	Overall packet size average
Feature#20	Active Mean	Mean active time of flow
Feature#21	Idle Mean	Mean idle time of the flow
Feature#22	SYN-to-FIN Ratio	Attack signature
Feature#23	Entropy Src IP	Source randomness score
Feature#24	Entropy Dst Port	Port usage variation
Feature#25	Connection Rate	mber of connections from source/sec

The Pseudocode explaining the inclusion of above mentioned 25 features as an input to the XG-Boost algorithm to classify the suspicious flow as DDoS or Benign is shown in Algorithm

3.

Algorithm 3: Pseudocode for the training of XG-Boost-25 Algorithm

Sr No.	ALGORITHM
1	# defining 25 features as an input to the XG-Boost algorithm def extract_25_features(flow_stats): features = [] features.append(flow_stats. duration) features.append(flow_stats.total_fwd_packets) features.append(flow_stats.total_bwd_packets) features.append(flow_stats.fwd_bytes) features.append(flow_stats.bwd_bytes) features.append(flow_stats.bytes_per_sec) features.append(flow_stats.packets_per_sec) features.append(np.mean(flow_stats.fwd_packet_lengths)) features.append(np.mean(flow_stats.bwd_packet_lengths)) features.append(np.mean(flow_stats.iat)) features.append(np.std(flow_stats.iat)) features.append(sum(flow_stats.fwd_iat)) features.append(sum(flow_stats.bwd_iat)) features. append(flow_stats.psh_flags) features. append(flow_stats.ack_flags) features. append(flow_stats.urg_flags) features.append(flow_stats.init_win_bytes_fwd) features.append(flow_stats.init_win_bytes_bwd) features.append(flow_stats.avg_packet_size) features.append(flow_stats.active_mean) features.append(flow_stats.idle_mean) features.append(flow_stats.syn_fin_ratio) features.append(flow_stats.entropy_src_ip) features.append(flow_stats.entropy_dst_port) features.append(flow_stats.conn_rate)
2	return np.array(features). reshape (1, -1)
3	# Training of XG-Boost on the 25 features input import XG-boost as xgb model = xgb.Booster() model.load_model("xgb_ddos_model.json") flow_features = extract_25_features(flow_stats) dtest = xgb.DMatrix(flow_features) pred_prob = model.predict(dtest)

4	# Apply threshold for binary class
	if pred_prob[0] > 0.5:
	# DDoS detected
	drop_flow(flow_id)
	Else:

```
allow_flow(flow_id)
```

Comments:

This code begins by importing the important library functions in the form of the xgboost library, initializing a Booster model, extracting 25 features from incoming network flow statistics, and converting them into an xgb. D-Matrix. The model predicts the probability of the flow being malicious. If the predicted probability exceeds 0.5, the flow is classified as a DDoS attack and dropped using drop_flow(flow_id); otherwise, it is allowed through using allow_flow(flow_id).

3.2.3 Working of the third Deep Learning Algorithm Graphical Neural Network (GNN) in the Proposed Algorithm:

When the flow is predicted malicious by the XG-Boot-25 algorithm, the SDN controller uses the services of the Graphical Neural Network (GNN) algorithm to detect whether the flow is part of a large distributed attack path or not. For this task, the GNN constructs a graph G (V, E) that is a subset of V and E. Where (V) represents all hosts (h1-h3, h4-h5), switches (S1-S5), and servers (h6 and h7). While (E) indicates Edges, communication paths, and statistical similarity (Time synchronization, common target, pool overlap, etc.).In GNN, each node and edge is labelled with entropy, packet rate, etc.GNN in the proposed framework is utilized to provide two important classifications: (1) node role (node as attacker, victim, and normal) and (2)Group Coordination(whether the flagged node is part of the Botnet structure). The GNN in the proposed framework can have three outputs: (1) isolated attacker (individual node is acting as an attacker and mitigation is performed only at this node). (2) Botnet detected (multiple coordinated nodes are launching the attack, so the whole network should be dropped out or mitigated as per the open flow rule, and (3) false positive (node is not malicious and

flow is permitted).

3.2.4 Implementation of Honeypot in the Proposed Algorithm:

Once the malicious flow is detected by the proposed multi-layer hybrid deep learning algorithms, the Pox controller installs the OpenFlow rules to redirect this malicious flow to the Honeypot segment (h7) via switch (S3) instead of damaging the banking server (h6) via switch (S2). The Honeypot can mimic the real banking platform, believing the attack that they are in a real banking server and keeping them engaged, and perform forensic analysis of their attack and forward all the relevant details to the SDN controller so that such a type of Botnet DDoS attack can easily be detected and immediately rejected without undergoing a multi-layer operational process. The complete working of the proposed hybrid multi-layer algorithm is explained in Fig.3.In this figure, the SDN Pox controller continuously monitors the flow statistics of the aggregated switch (S1) and forwards the data set in the form of ($X\epsilon R^{10\times6}$) to LSTM-AE, which calculates the fault function ($MSE = \frac{1}{n}\sum_{i=1}^{n}(x_{i-} x_i^{-2})$ and compare it with the allowed threshold (θ_{AE}). If reconstruction error (ϵ) = (MSE) > $\theta_{AE,}$ then the flow is considered suspicious and forwarded to the XG-Boost-25 algorithm that uses 25 features as input to classify the flow as normal or DDoS attack. After that DDoS attack-oriented flow is forwarded to GNN to detect that the DDoS attack is generated by an isolated node or by using a Botnet mechanism. Upon detection of DDoS attacks (by isolated node or Botnet mechanism) with the proposed hybrid multi-layer algorithm, the mitigation is performed; not just by rejection of the flow, but to have detailed attacker deception and forensic logging, the malicious flow is forwarded to an isolated Honeypot environment.

3.3 Mathematical Modelling of the proposed hybrid multi-layer deep learning algorithm to mitigate Botnet DDoS attacks

This research proposes a multi-layer Software Define Networking (SDN)-Pox controller-based structure installed on Mininet featuring an intelligent, real-time a hybrid deep learning framework for DDoS attacks mitigation based on three algorithms and Honeypot implementation. The mathematical model of the whole hybrid proposed framework is shown in Eq. (4-12) and is as follows:

Defining parameters:

F_T is the multi-variable flow vector for time(t) and other differential parameters used in the mathematical design of the proposed frame, as explained in Tab.4

TABLE 4: Defining differential parameters used in the mathematical design of the proposed frame

Sr.NO	Features per Step	Description
1	$(X=F_T\text{-}\Delta t\ldots\ldots, F_T)$	Input sequence to LSTM-AE
2	$\hat{X}$	reconstructed sequence by LSTM-AE
3	$\epsilon(X)$ and $D(X)$	Both are encoder and decoder functions
4	$\epsilon = (\lvert X - \hat{X}\rvert^2)$	reconstruction error
5	θ_{thres}	error threshold for anomaly detection
6	$\Phi(F_T)$	feature vector (25) extracted
7	y (0,1)	ary class label (1) for DDoS and 0 for benign flow
8	C_{XGB}	trained XG-Boost classifier
9	G (V, E)	dynamic host communication graph

10	H_v	hidden state of node v€ V in GNN
11	A	adjacency matrix
12	Z= GNN (A, H)	GNN output
13	M_{mit}	mitigation policy function in the Pox controller

LSTM-AE Modelling:
Input Data set function:
(X=F_T-Δt....., F_T) Is the time series window of flow with features (F_T€ R^d) to LSTM-AE
Encoder ϵ(X):

$$h_t = LSTM_E\ (F_T;\ W_e) \quad (4)$$

Decoder D(X):

$$\widehat{F_T} = LSTM_d\ (h_t;\ W_d) \quad (5)$$

Reconstruction error

$$\epsilon = \frac{1}{T}\sum_{t=1}^{T}(|X - \hat{X}|)2 \quad (6)$$

If $\epsilon > \theta_{thres,}$ then the flow is considered malicious and forwarded to the XG-Boost-25 algorithm.

XG-Boost-25 Modelling:
Input Data set function:
$\Phi(F_T)$ € R^{25} is the feature vector (25) for the XG-Boost -25 algorithm
Classifier function:

$$y = C_{XGB}(\Phi(F_T)) = \sum_{k=1}^{K} f_k(\Phi(F_T)) \quad (7)$$

Where (f_k € f), K= number of trees and f_k = regression tree
Regression loss:

$$L(\theta) = \sum_i l(yi, \hat{y}_i) + \sum_k \Omega(fk) \quad (8)$$

$$\Omega(f) = \Upsilon T + \frac{1}{2}(|W|)^2 \quad (9)$$

T= no of leaves, W= leaf weight and L= loss

GNN Modelling:

Defining parameters:

Nodes= source IPS or host IDS, Edge= flow interactions, Edge weight = temporal correlation frequency with flow volume similarity, N(v) = neighbour of v, $W_{(l)}$ = trainable weight matrix, and $G_v^{(0)}$ = initial mode features

Message Passing (for each layer l and node v):

$$H_v(loss+1) = \sigma\left(\sum_{u \in N(v)} \frac{1}{\sqrt{dv.du}} W^{(l)} h_u^{(l)}\right) \quad (10)$$

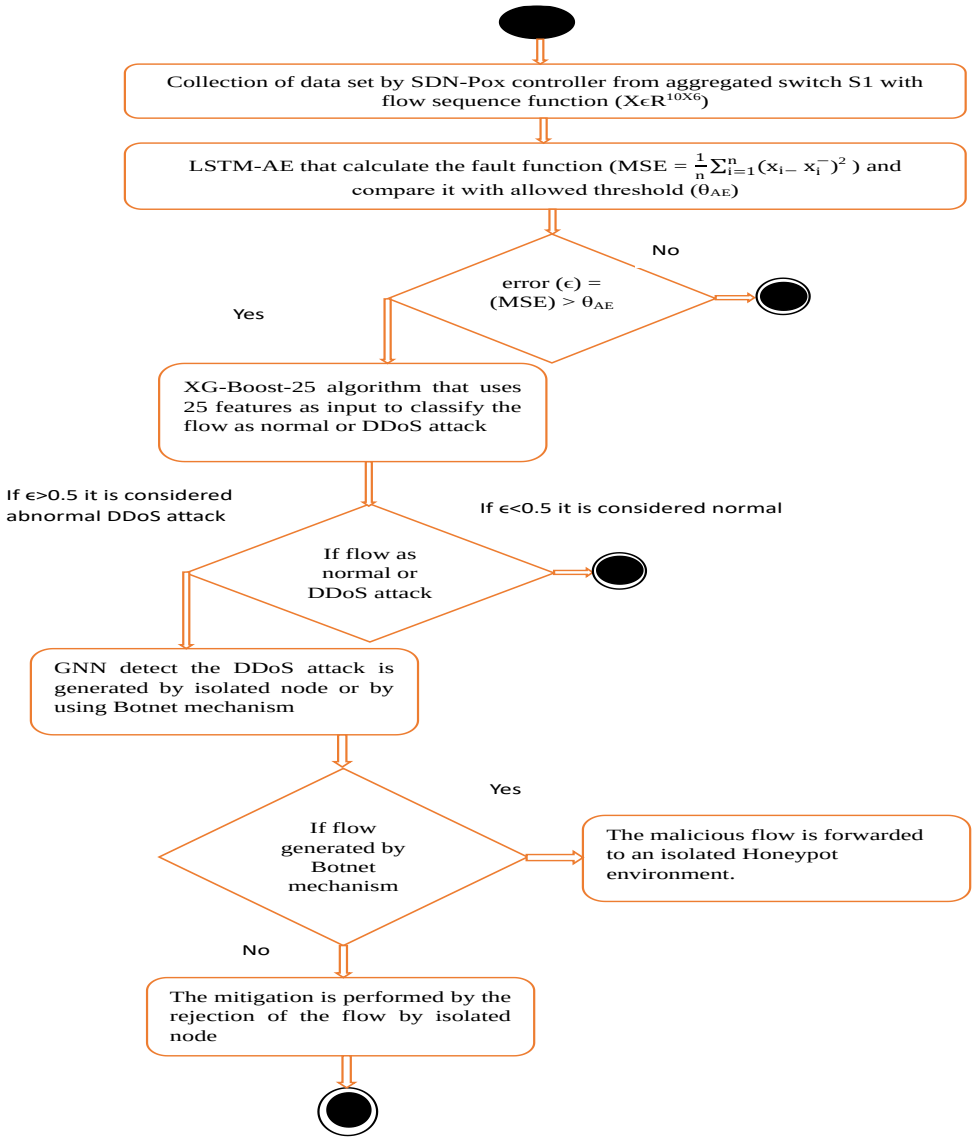

Figure 3: UML flow diagram of the proposed hybrid multi-layer algorithm

SDN Mitigation policy Via Pox controller based on Z_v (GNN trust score) and τ (trust threshold):

Flow tagging:

If the proposed framework provides the result for any flow as (LSTM -AE detects flow as suspicious, XG-Boost-25 further detects this flow as DDoS attack, and GNN rates this flow as Botnet affiliated then:

$$M_{mit}\ (flow\text{- }id) = (y=1 \text{ and } Z_v > \tau)\ flow\ is\ forwarded\ to\ Honeypot \quad (11)$$

OR
M_{mit} *(flow- id) = (y=1 and* $Z_v \leq \tau$*) rate limit (1Mbps)*
OR
If y=0, then allow the flow

Composite Evaluation Matrix:

$$P = \sigma_1 . A + \sigma_2 \left(1 - \frac{Ld+Lm}{T}\right) + \sigma_3\left(\frac{B}{Bmax}\right) + \sigma_4 (1-P_L) \quad (12)$$

Subject to:
A= Accuracy
L_d = detection latency
L_m= mitigation latency
T= total window
B= retained bandwidth
PL= Packet loss
σ_i = weight sum (equal to 1), here ($\sigma_{1-3} = 0.3$ and $\sigma_{2-4} = 0.2$)

4 Results and Discussion

In order to prevent the digital financial application main/centralized server (h6: housing in the application layer of our topological arrangement as shown in Fig.2) from modern-day Botnet-oriented DDoS attack, the proposed hybrid deep learning based SDN-Pox controller is designed on the Mininet environment. The SDN-Pox controller provides mitigation from Botnet-oriented DDoS attacks by utilizing a multi-layer framework containing three deep learning algorithms (LSTM-AE, XG-Boost, and GNN).To check the efficiency of proposed framework the two test cases are explained in this section and performance of proposed framework is calculated in each case by using values of (8) deterministic matrices (bandwidth: B_{dava} in Mbps, percentage packet loss:%P_{Loss}, API response time: $T_{API\ \_Response\ time}$, connection failures: $T_{connection\_failures}$, detection time: $T_{Detection}$, Mitigation time: $T_{Mitigation}$, latency: $LT_{(ms)}$ in ms, and retransmission: $RE_{transmission}$). We have developed a scenario in Mininet that during total simulation of 20sec the first 10 sec the core server (h6) in our design topology was not attacked attacker clients (h1-h3) and flow was normal and legitimate and at 11 sec the botnet based attack is launched by hosts (h1-h3) and then this flow are detected

and mitigated by SDN-Pox controller with proposed hybrid multi-layer deep algorithm (LSTM-AE, XG-Boost-25, GNN). The time sequence scenario is intentionally selected to see how much time ($T_{Detection}$ and $T_{Mitigation}$) the SDN-Pox controller with the proposed hybrid multi-layer framework takes to detect and mitigate the Botnet-oriented DDoS attack. The Pseudocode of this time sequence scenario is explained in Algorithm 4.

Algorithm 4: Time sequence scenario of the proposed topological structure in Mininet

	ALGORITHM
1	# Define components of user-defined topology in the Mininet environment
	h1, h2, h3: Botnet oriented attacker clients
	h4–h5: legitimate clients
	h6: core banking API server (target)
	h7: honeypot server
	S1–S5: OpenFlow switches
2	Launch the POX controller with the proposed hybrid multi-layer deep learning module.
3	Simulate the Mininet network.
4	# Generating the Traffic scenario
	For time t = 0 to 10 sec:
	h1–h3: not operational
	h4–h5: Continue normal banking transactions to h6
	At t = 11 sec:
	h1–h3: Launch botnet-based SYN/HTTP/Slowloris DDoS attacks on h6
5	# Flow Monitoring via POX
	Every Δt = 1 sec:
	For each active flow f:
	Request flow stats from OpenFlow switches
	Extract features (packet rate, protocol, flags, etc.)
	Append to time-series window X for LSTM-AE
	# LSTM Layer
	For each time window X:
	Reconstruct output: X_hat = LSTM_AE. predict(X)
	Calculate error: ε = MSE (X, X_hat)
	If ε > threshold_ε:
	Mark flow f as anomalous
	Send to XGBoost-25 classifier
	# XG-Boost-25 Layer
	Input: Top 25 features from anomalous flow f

```
Output: Label y ∈ {benign, attack_type}
If y == attack:
Mark F as malicious
Forward source IP and graph context to GNN
Else:
Allow flow
# GNN Layer
Construct graph G:
Nodes = source hosts
Edges = communication to common destinations
For node v in G:
 Apply GNN(v) → output trust score Z_v
If Z_v < trust_threshold:
 Confirm the attacker is part of a botnet
 Notify mitigation module
# SDN-Pox mitigation
For each flow f with a confirmed attack:
 If Z_v < trust_threshold:
Inject OpenFlow rule to:
Drop flow
 OR
Redirect flow to honeypot (h7)
 Else if single host:
Apply rate-limit (1 Mbps)
# extracting the parameters of deterministic matrices
 Log for each 1-second interval:
Detection time (from t_attack to anomaly flag)
Mitigation time (from detection to flow rule injection)
Packet loss, retained bandwidth
TCP failures, API response time
# scenario termination
 At t = 20 sec:
Stop attack scripts
Stop Mininet and Pox
Generate comparative results and J-index
```

Comments:

The bandwidth and data transfer are calculated by using the Iperf utility. The percentage

packet loss is calculated by using the ping command of the Iperf tool in User Datagram Protocol (UDP) mode and is also verified by Eq.13.The latency is calculated by using the use of ping command (Iperf3-i1) and measured as the average of Round-Trip Time (RTT). The retransmission factor is calculated by using the Iperf tool (iperf3--, verbose _TCP stats) and mathematically verified by the count of Transmission Control Protocol (TCP) segments retransmitted due to Acknowledgment (ACK) delays. The API response time is calculated by using curl commands (ab, Curl-w, application logs) and mathematically verified by using Eq.14.The TCP connection failures are calculated by using the (netstat, Iperf 3) tool and mathematically verified by the count of refused connections. The T_{Detect} and $T_{Mitigation}$ are calculated by using Eqs. (15 and 16).

$$\% \ Packet\ loss = (P_{sent} - P_{received}) \times 100 \quad (13)$$

where P_{sent} and $P_{received}$ are the number of packets sent or received

$$Response\_Time = T_{first\_byte} - T_{request\_sent} \quad (14)$$

where T_{first_byte} (time for first byte sent) and $T_{request\_sent}$ (time for total request sent)

$$T_{detect} = T_{collect} + T_{features} + T_{inference} \quad (15)$$

where:

$T_{collect}$ is the flow statistics polling interval from switches (1-2 secs via OpenFlow)

$T_{features}$ are the time to extract flow features (10-30ms)

$T_{collect}$ is the combined inference time of LSTM-AE, XG-Boost-25, and GNN (10-

25ms)

$$T_{mitigation} = T_{decision} + T_{rule\_install} + T_{switch\_update}$$

where:

$T_{decision}$ is the time to choose the mitigation policy (10-15 ms) *(16)*

$T_{rule\_install}$ is the time for the controller to push flow rules (5-10 ms)

T_{switch_update} is the time for the switch to enforce rules (5-15ms)

4.1 (Case-1): Measuring deterministic metrics of core-API (h6) of the designed topological arrangement with the proposed framework during legitimate and Botnet flow

The user-defined topological structure, as explained in Fig.2, has a core server of (h6) providing necessary financial data to all the clients in the client layer via the communication layer switch (S1). The hosts (h1-h3) act as Botnet-oriented malicious sources that can launch a DDoS attack and exhaust the resources of the core server (h6) and making its access limited or unreachable to legitimate users (h4-h5). In this case (Case-1), to compare the performance and effectiveness of the proposed framework, the time sequence scenario as explained in Algorithm 4 is implemented on Mininet, and the parameters of deterministic metrics are calculated via the Iperf utility tool and further verified using Equations (Eq.13-16). The complete parameters of the deterministic metric (B_{dava}, $\%P_{Loss}$,$T_{API\_Response\ time}$, $T_{connection\_failures}$, $T_{Detection}$, $T_{Mitigation}$, T_{data_rate}, and $RE_{transmission}$) obtained with the Iperf testing tool kit and mathematical equations (Eq.13-Eq.16) are shown in Tab.5.

TABLE 5: Parameters of the deterministic metric (B_{dava}, $\%P_{Loss}$, $T_{API\_Response\ time}$, $T_{connection\_failures}$, $T_{Detection}$, $T_{Mitigation}$, T_{data_rate}, and $RE_{transmission}$) using the Iperf utility.

Time (sec)	Bd_{ava} (Mbps)	% P_{Loss}	LT (ms)	RE($_{Transmission}$)	$T_{API\_response}$ (ms)	$T_{connection\_failure}$	T_{data_rate} (Mbps)	SDN-Pox controller Status	SDN-Pox Mitigation Status
0-10	850	0.1	15	5	180	0	840	Sensed as normal via LSTM-AE control layer	No action
11	120	13.5	540	310	2200	40	110	Sensed flow suspicious	No action
11.190	810	1.7	32	14	290	2	790	DDOS detected via Botnet	Honeypot segmentation
11.190-20	810	1.7	32	14	290	2	790	Sensed normal again via LSTM-AE control	No action

In Tab.5, it is clear that during total simulation of 20sec the first 10 sec the core server (h6) was not attacked by the attacker clients (h1-h3) and flow was normal and legitimate and at 11 sec the botnet based attack is launched by hosts (h1-h3) and then this flow is detected and mitigated by SDN-Pox controller with proposed hybrid multi-layer deep algorithm (LSTM-AE, XG-Boost-25, GNN) within 190ms. The parameters (B_{dava},$\%P_{Loss}$, $T_{API\_Response\ time}$, $T_{connection\_failures}$, $T_{Detection}$, $T_{Mitigation}$, T_{data_rate}, and $RE_{transmission}$) of deterministic metrics that were reduced during the Botnet DDoS attack at 11sec is are again amplified to the normal flow values within 190ms time by SDN-Pox controller using the proposed hybrid multi-layer framework. The Gnu-plot is used to visualize the flow patterns of different parameters of deterministic metrics. The Fig.4 represents the flow pattern of (B_{dava} and T_{data_rate} in Mbps) in the proposed scenario as explained in Algorithm 4. The Fig.5 represents the flow pattern of (LT and $T_{API\_response}$ in ms) in the proposed scenario as explained in Algorithm 4. Fig.6 represents the flow pattern of (% P_{Loss}) and Fig.7 represents the flow pattern of

($T_{connection\_failure}$) in the proposed scenario as explained in Algorithm 4. In Fig. (4-7), to have better visualization, the values of time in seconds on the x-axis start from 10 sec to 15 sec (because from 0-10 sec the values are same due to normal flow scenario designed in Algorithm 4 and same is case from 15-20 sec) with a time stamp of 0.5 seconds to better judge the performance of the proposed hybrid multi-layer framework used by SDN-Pox controller.

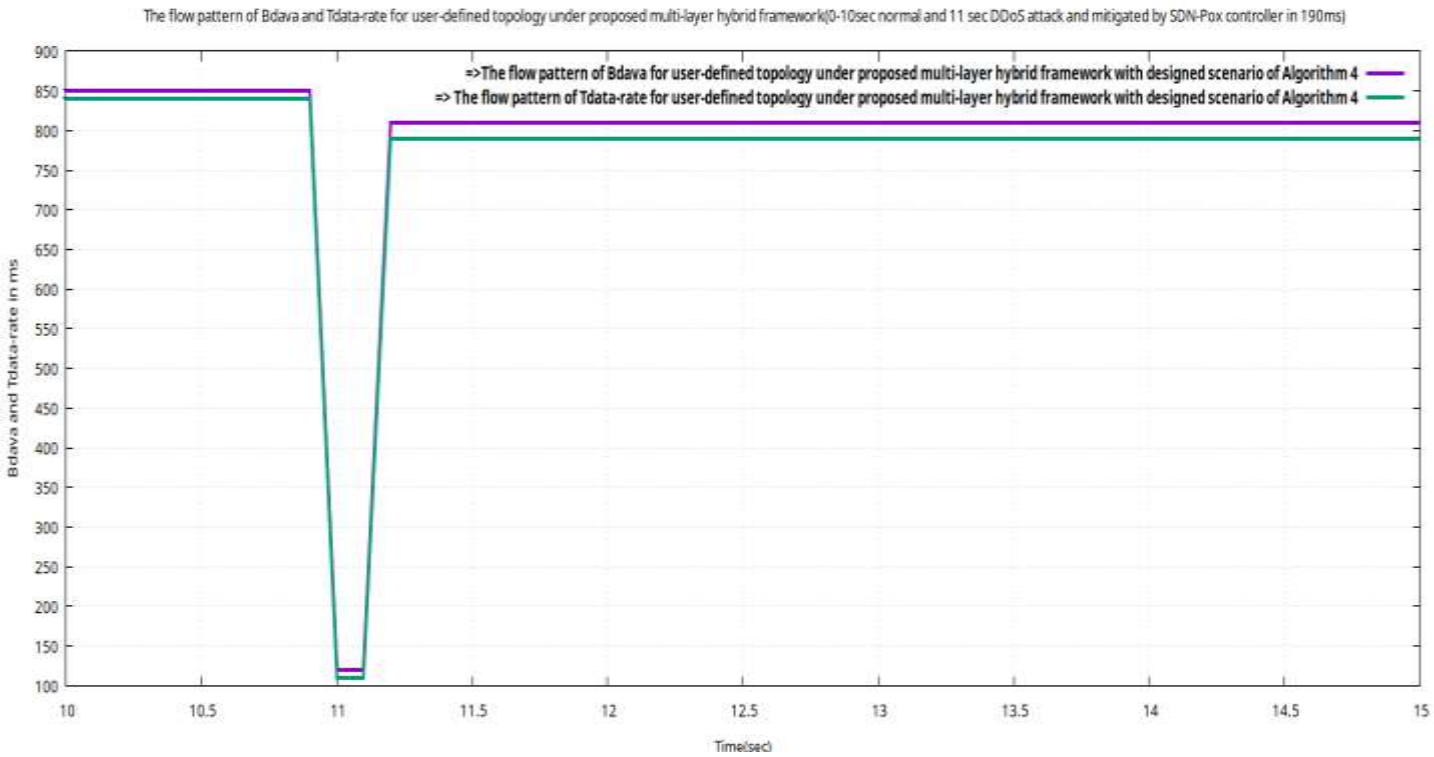

Figure 4: Flow pattern of (B_{dava} and T_{data_rate} in Mbps) in the proposed scenario as explained in Algorithm 4

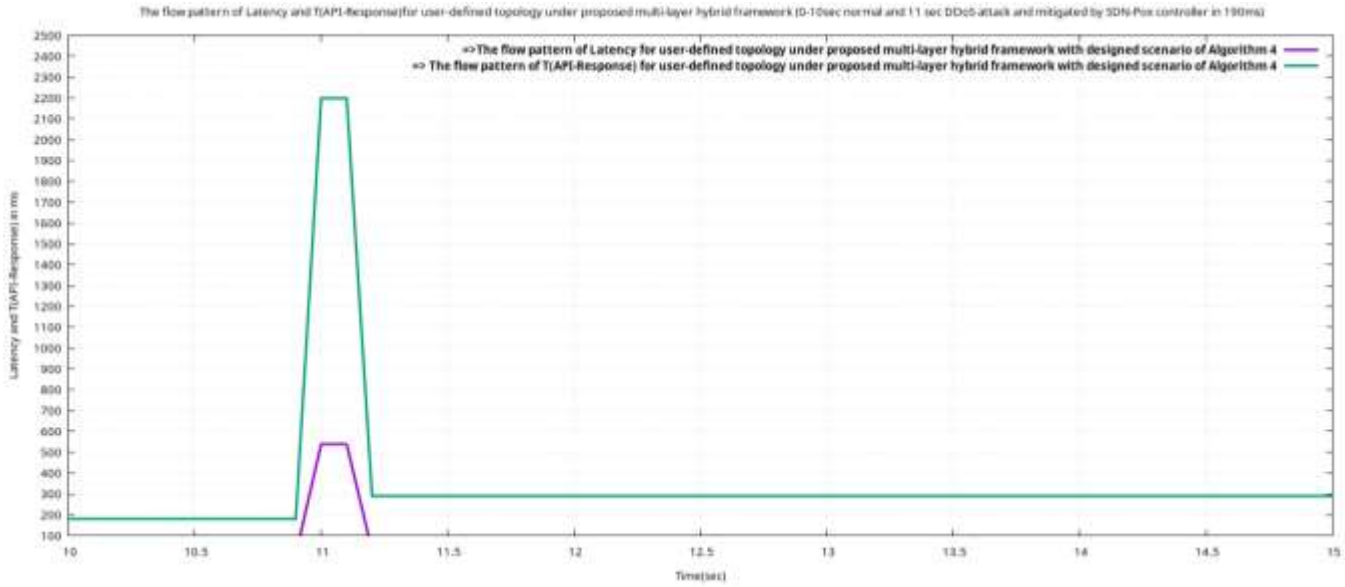

Figure 5: Flow pattern of (LT and $T_{API\_response}$ in ms) in the proposed scenario as explained in Algorithm 4

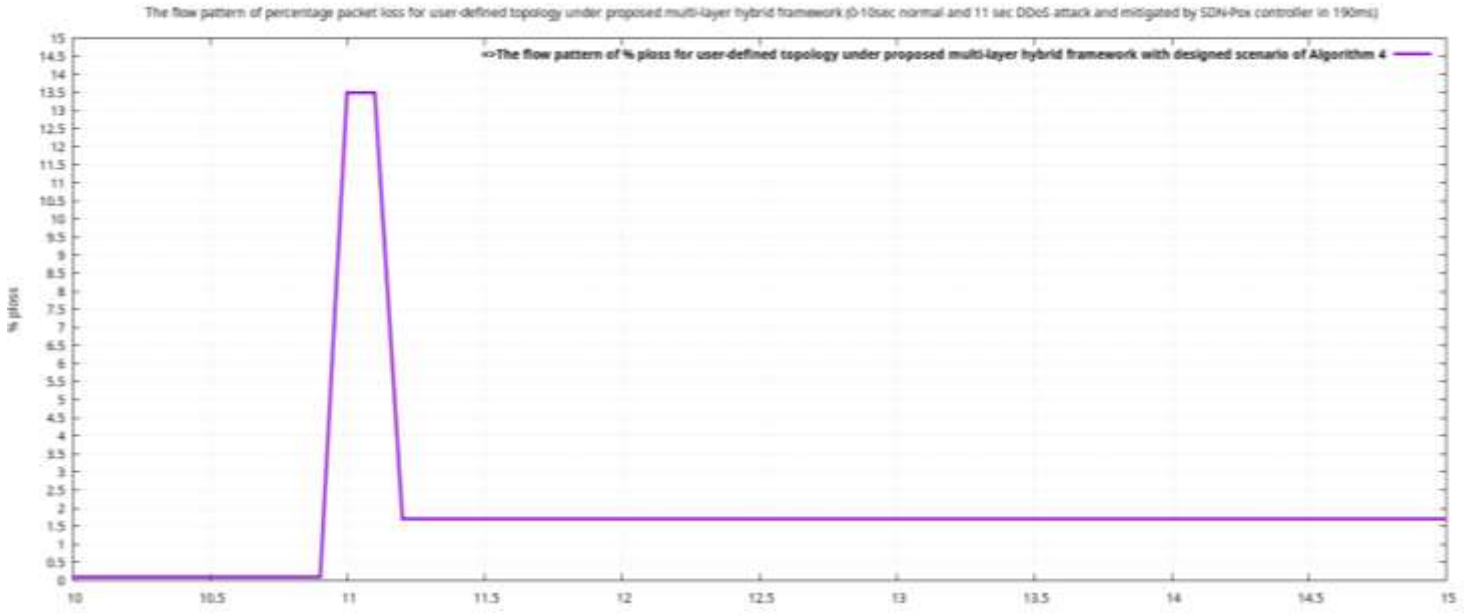

Figure 6: The flow pattern of (% P_{Loss}) in the proposed scenario as explained in Algorithm 4

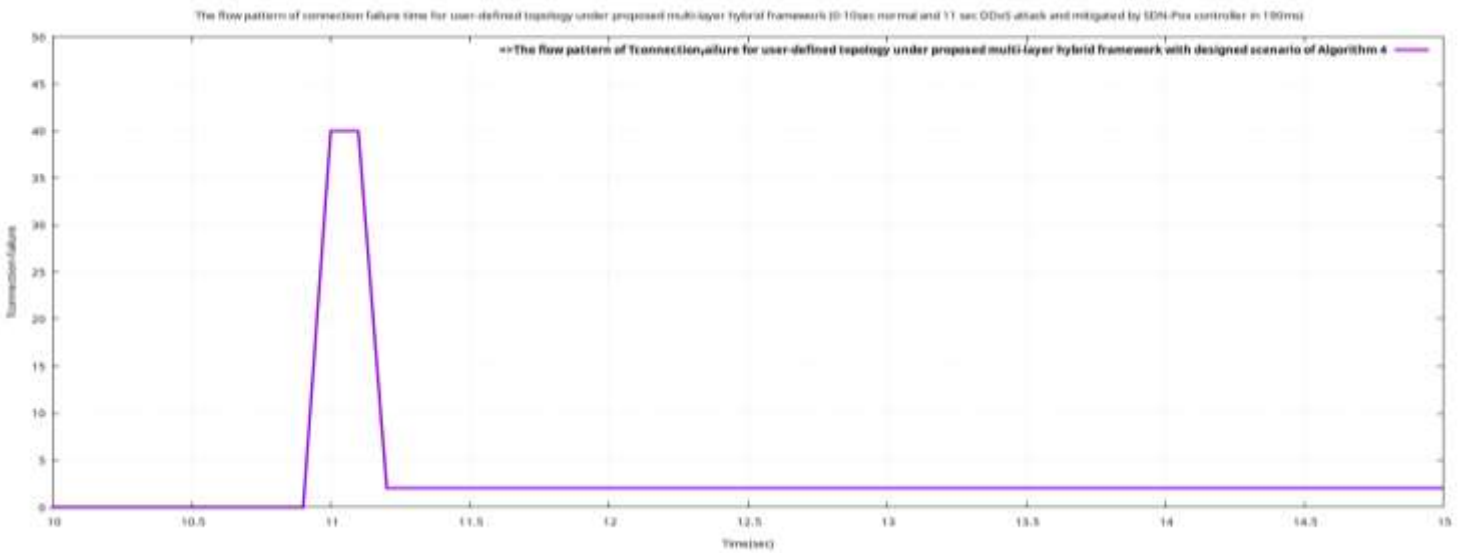

Figure 7: The flow pattern of ($T_{connection\_failure}$) in the proposed scenario as explained in Algorithm 4.

4.1.5 Summary of Case-1:

The user-defined network topology is designed on Mininet with SDN-Pox controller using the proposed multi-layer hybrid framework under a time sequence scenario as explained in Algorithm 4. The results of (Case-1) about different parameters of deterministic metrics during time sequence scenario prove that the proposed framework effectively mitigates the Botnet-oriented DDoS attack, and the value of all parameters of deterministic metrics is adjusted back to their values during normal flow. The value of (Bd_{ava}) increased from 120Mbps to 810 Mbps. The same is the case for T_{data_rate} from 110 Mbps to 790 Mbps. The latency in ms is reduced from 540 ms to 32 ms. The $T_{API\_response}$ in ms is also reduced from 2200ms to 290ms with SDN-POX honeypot mitigation.

4.2 (Case-2: Comparison of Proposed hybrid multi-layer deep learning framework for Botnet-oriented DDoS attack with traditional mitigation methods)

In this case, the comparative analysis is performed by evaluating the performance of the

SDN-POX controller integrating a proposed hybrid multi-layer framework for mitigation of Botnet-oriented DDoS attacks with traditional mitigation methods (CNN-GRU, Transformer-IDS, Random Forest, and Signature-based Intrusion Detection System _IDS). For this task the all these frameworks are made functional on the user-defined topological structure as shown in Fig.2, and DDoS attacks are created by Hosts (h1-h3), and the values of different parameters (B_{dava}, $\%P_{Loss}$, $T_{API\_Response\ time}$, $T_{\ connection\_failures}$, $T_{Detection}$, $T_{Mitigation}$, T_{data_rate}, and $RE_{transmission}$) of deterministic metrics for the core server (h6) are calculated in both the proposed framework and traditional mitigation methods. The values of parameters of the deterministic metric (B_{dava}, $\%P_{Loss}$, $T_{API\_Response\ time}$, $T_{\ connection\_failures}$, $T_{Detection}$, $T_{Mitigation}$, T_{data_rate}, and $RE_{transmission}$) in all above-mentioned scenarios obtained with the Iperf testing tool kit and mathematical equations (Eq.13-Eq.16) are shown in Tab.6.

TABLE 6: Comparative analysis data of different parameters of the deterministic metric (B_{dava}, $\%P_{Loss}$, $T_{API\_Response\ time}$, $T_{\ connection\_failures}$, $T_{Detection}$, $T_{Mitigation}$, T_{data_rate}, and $RE_{transmission}$) using the Iperf utility

Sr.No	Technique utilized	$T_{detection}$ (ms)	$T_{Mitigation}$ (ms)	Bd_{ava} (Mbps)	% P_{Loss}	$T_{API\_response}$ (ms)	$T_{connection\_failure}$	T_{data_rate} (Mbps)
1	Proposed SDN–Hybrid (LSTM-AE + XGBoost-25 + GNN	160	30	810	1.7	290	2	790
2	CNN–GRU	230	60	760	3.2	350	4	720
3	Transformer-IDS	200	45	780	2.8	330	3	740
4	Random Forest	550	200	390	12.4	850	17	370
5	Signature-IDS	850	300	290	18.5	1400	32	260

In Tab.6, it is clear that during the total simulation of 20 seconds, the core server (h6)

deterministic metrics by SDN-POX controller with the proposed hybrid multi-layer deep algorithm (LSTM-AE, XG-Boost-25, GNN) are far superior to traditional mitigation methods. The comparison of the flow pattern of parameters of the deterministic metric (B_{dava}, $T_{API\_Response\ time}$, $T_{Detection}$, $T_{Mitigation}$) using Gnu-plot is shown in Fig. 8(a-d) for both proposed and traditional mitigation methods. In Fig.8(a), the value of ($T_{detection}$) is minimum (160ms) in the proposed hybrid multi-layer framework as compared to traditional mitigation methods (CNN-GRU, Transformer-IDS, Random Forest, and Signature-based Intrusion Detection System _IDS)

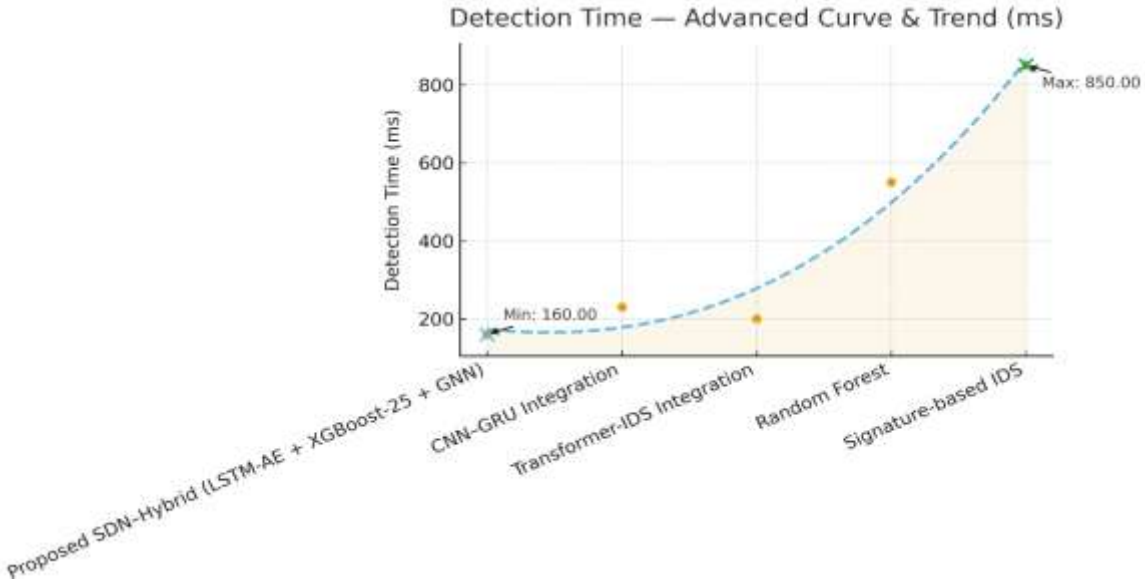

Figure 8(a): The comparison of the flow pattern of ($T_{detection}$) in proposed and traditional mitigation frameworks.

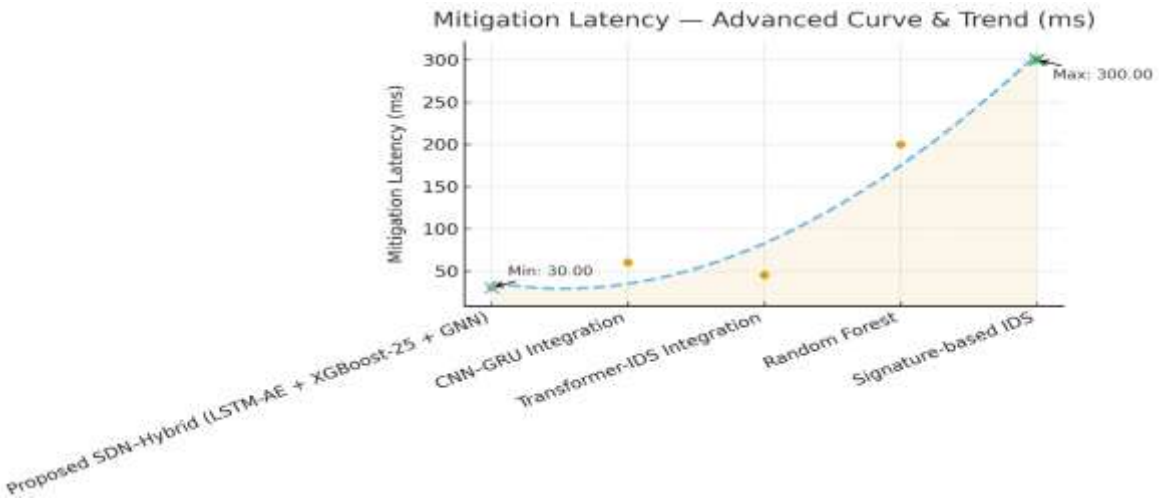

Figure 8(b): The comparison of the flow pattern of ($T_{mitigation}$) in proposed and traditional mitigation frameworks.

In Fig.8(b), the value of ($T_{mitigation}$) is minimum (30ms) in the proposed hybrid multi-layer framework as compared to traditional mitigation methods (CNN-GRU, Transformer-IDS, Random Forest, and Signature-based Intrusion Detection System _IDS). In Fig.8(c), the value of $T_{API\_Response\ time}$ is minimum (290 ms) in the proposed hybrid multi-layer framework as compared to traditional mitigation methods (CNN-GRU, Transformer-IDS, Random Forest, and Signature-based Intrusion Detection System _IDS).

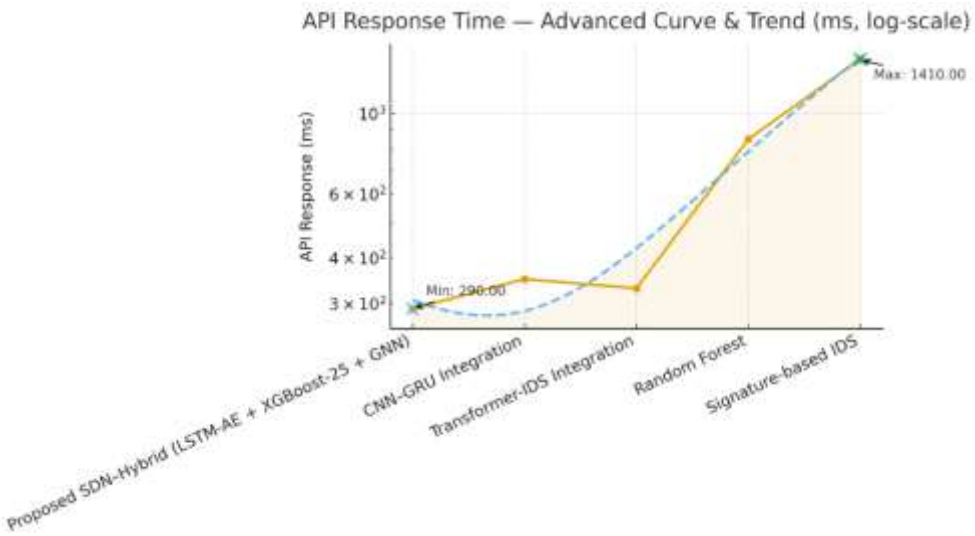

Figure 8(c): The comparison of the flow pattern of $T_{API\_Response\ time}$ in proposed and traditional mitigation frameworks

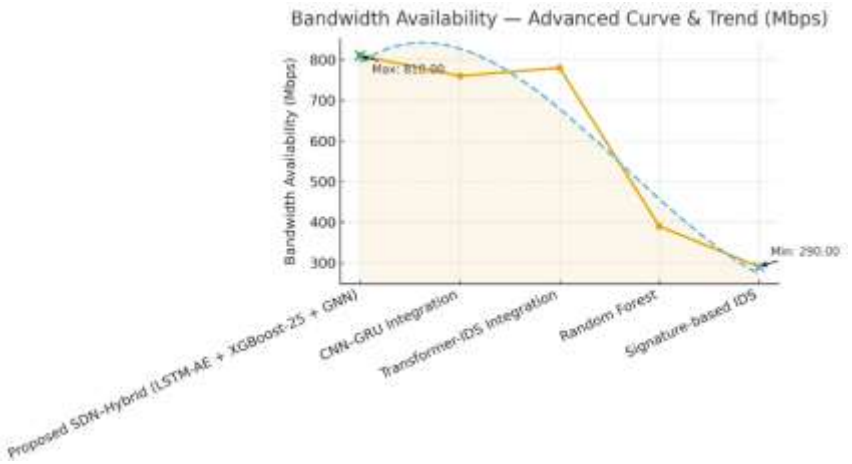

Figure 8(d): The comparison of the flow pattern of B_{dava} in the proposed and traditional mitigation frameworks

In Fig.8(d), the value of B_{dava} is maximum (810 Mbps) in the proposed hybrid multi-layer framework as compared to traditional mitigation methods (CNN-GRU, Transformer-IDS, Random Forest, and Signature-based Intrusion Detection System _IDS). The comparative analysis of factor (J) in the proposed hybrid multi-layer framework with traditional mitigation methods is shown in Tab. 7. The factor (J) is calculated by (Eq. A) in the introduction section and defined as the overall degradation or risk to system performance under DDoS conditions, combining multiple metrics into a single scalar value. The higher the value of the function (J), the slower and more sluggish will be the response of the mitigation framework. The value of the weights of factor J, as explained in Eq. A are (α=0.3, β = 0.2, γ= 0.1, δ= 0.15, ε= 0.15, ζ= 0.1)

TABLE 7: Comparative analysis data of function (J) using the Iperf utility

Sr.No	**Technique utilized**	**A (detection Accuracy)**	**L_d (ms)**	**L_m (ms)**	**% P_{Loss}**	**Bd_{ava} (Mbps)**	**$T_{API\_response}$ (ms)**	**$T_{connection\_failure}$**	**J**
1	Proposed framework	0.989	160	30	1.7	810	290	2	78.7036

2	Random Forest	0.924	550	200	12.4	390	850	17	259.2276
3	Signature-IDS	0.902	850	300	18.5	290	1400	32	413.239

In Tab.7, it is clear that during the total simulation of 20 seconds, the value of factor (J) in the SDN-POX controller with the proposed hybrid multi-layer deep algorithm (LSTM-AE, XG-Boost-25, GNN) is less than traditional mitigation methods (Random Forest and Signature-IDS). The comparison of the flow pattern of factor (J) using Gnu-plot is shown in Fig. 9 for both proposed and traditional mitigation methods.

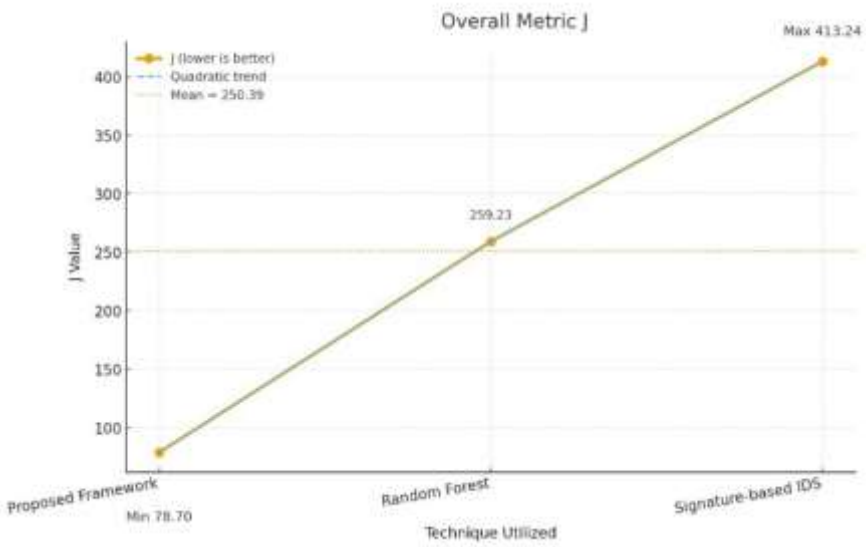

Figure 9: The comparison of the flow pattern of factor (J) in proposed and traditional mitigation scenarios.

Summary of Case 2:

The user-defined network topology is designed on Mininet with the scenario explained in Algorithm 4 and evaluated separately for both the proposed framework and traditional mitigation methods. The different parameters of deterministic metrics are calculated for the core server (h6) in both (proposed and traditional) scenarios. SDN-Pox controller using the proposed multi-layer hybrid framework outperformed the traditional mitigation methods for

managing the Botnet-oriented DDoS attack in just 190ms. The results of factor (J) are also minimum as per research objectives in comparison to traditional mitigation methods (Random Forest and Signature-IDS).

5 Conclusion

In this research, the proposed hybrid multi-layer deep learning framework using (LSTM-AE, XG-Boost, GNN) in the SND-Pox controller control layer for real-time mitigation of the evolving sophistication of Distributed Denial of Service (DDoS) attacks targeting financial systems using protocol-aware botnets and covert Layer 7 routes. With the implementation of the proposed framework, the overall system resilience is further strengthened by deceptive mitigation employing honeypot redirection, which also makes forensics logging and trust-aware isolation possible. The proposed structure, which was executed using the POX controller in a Mininet-based banking environment Modeling, showed outstanding bandwidth retention (810 Mbps), quick mitigation (30 ms), and quick detection (160 ms) with a packet loss rate of just 1.7%. The proposed method outperformed Random Forest and signature-based intrusion detection systems in every important statistic, including the J-index composite score. This proves the ease with which it can detect and stop sophisticated multi-source DDoS attacks without interfering with the operation of trustworthy financial services. The possible future developments of this research include implementing the system in hardware-accelerated SDN environments, incorporating federated learning for distributed detection, and expanding the model to identify insider threats and data exfiltration efforts. In the end, the suggested framework provides a strong and perceptive defence layer to protect next-generation financial infrastructures from the constantly changing cyberattack scene.

Acknowledgment: The authors gratefully acknowledge the support and facilities provided by the Telecommunication Research Center of the Electrical Engineering Department of Bahauddin Zakariya University, Multan, Pakistan, throughout this research endeavor.

Funding Statement: The author(s) received no specific funding for this study.

Author Contribution: The author's contributions to this paper are as follows: study conception and design: K.T. Mehmood; data collection: K.T. Mehmood; analysis and interpretation of results: K.T. Mehmood; draft manuscript preparation: K.T. Mehmood, R.Iqbal . All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: Data with the corresponding author can be provided upon appropriate request.

Ethics Approval: Not applicable.

Conflict of Interest: The authors declare that they have no conflicts of interest to report regarding the present study

References

1. Yoachimik O. Cloudflare mitigates 26 million requests per second DDoS attack: The Cloudflare Blog. Available from: https://blog.cloudflare.com/26m-rps-ddos/. [Accessed 2022].
2. Firch J. Mantis Botnet: The Largest DDoS Attack Ever Mitigated: PurpleSec (Cyber Attacks Blog). Available from: https://purplesec.us/breach-report/mantis-botnet/purplesec.us. [Accessed 28 April 2024].
3. Pardue L, Desgats J. HTTP/2 Rapid Reset: deconstructing the record-breaking attack: The Cloudflare Blog. Available from: https://blog.cloudflare.com/technical-breakdown-http2-rapid-reset-ddos-attack/. [Accessed 2023].
4. Snellman J, Iamartino D. How it works: The novel HTTP/2 'Rapid Reset' DDoS attack: Google Cloud Security Blog. Available from: https://cloud. google. Com/blog/products / identity-security/how-it-works-the-novel-http2-rapid-reset-ddos-attack. [Accessed 2023].
5. FS-ISAC, Akamai. DDoS Attackers Increase Targeting of Global Financial Sector, According to FS-ISAC and Akamai Report.Available from: https://www.fsisac.com/newsroom/ddos-attackers-increase-targeting-of-global-financial-sector. [Accessed 2025]
6. SC Staff. Pro-Russian hacktivists intensify DDoS attacks on Dutch orgs, SC Media (Cyber-Risk-Alliance). Available from: https://www.scworld.com/brief/pro-russian-hacktivists-intensify-ddos-attacks-on-dutch-orgs. [Accessed 2025]
7. Asokan A. Danish Banks Are Targets of Pro-Russian DDoS Hacking Group, Bank Info Security, Available from: https://www.bankinfosecurity.com/danish-banks-targets-pro-russian-ddos-hacking-group-a-20902. [Accessed 2023]
8. Nexusguard. Why Firewalls are Inadequate as Standalone DDoS Solutions, Nexusguard Blog. Available from: https://www.nexusguard.com/blog/why-firewalls-are-inadequate-as-standalone-ddos-solutions. [Accessed 18-Dec-2023]

9. Indusface. How to Stop DDoS Attacks: 17 Best Practices to Prevent DDoS, Indusface Blog. Available from:https://www.indusface.com/blog/best-practices-to-prevent-ddos-attacks/. [Accessed 2023]
10. Ashfaq F, Wasim M, Shah MA, Ahad A, Pires IM. Enhancing Security in 5G Edge Networks: Predicting Real-Time Zero Trust Attacks Using Machine Learning in SDN Environments. Electronics.2025; vol. 12(6), no. 6. https://doi.org/10.3390/electronics12061392.
11. Apostu AM et al. Detecting and Mitigating DDoS Attacks with AI: A Survey. arXiv preprint 2018. https://arxiv.org/html/2503.17867v1
12. Haseeb-ul-Rahman RMA et al. High-Speed Network DDoS Attack Detection: A Survey. Electronics 2023; vol. 12(5): p. 3112. Doi: 10.3390/s23156850
13. Arvind T, Radhika K. XGBoost Machine Learning Model-Based DDoS Attack Detection and Mitigation in an SDN Environment. Int. J. Engineering Trends and Technology 2023; vol. 71(2): pp. 349–361. Doi: 10.14445/22315381/IJETT-V71I2P237
14. Lo WW, Kulatilleke GK, Sarhan M, Layeghy S, Portmann S. XG-BoT: An explainable deep graph neural network for botnet detection and forensics. Internet of Things (Elsevier).2023; vol. 22: p. 100708.https://doi.org/10.48550/arXiv.2207.09088
15. Mozo A, Karamchandani A, de la Cal L, Gómez-Canaval S, Pastor A, Gifre L. A Machine-Learning-Based Cyberattack Detector for a Cloud-Based SDN Controller. Applied Sciences. 2023; 13(8):4914. https://doi.org/10.3390/app13084914
16. Lachekhab F, Benzaoui M, Tadjer SA, Bensmaine A, Hamma H. LSTM-Autoencoder Deep Learning Model for Anomaly Detection in Electric Motor. Energies. 2024; 17(10):2340. https://doi.org/10.3390/en17102340
17. Yu T, Xin Y, Zhang C. HoneyFactory: Container-Based Comprehensive Cyber Deception Honeynet Architecture. Electronics. 2024; 13(2):361. https://doi.org/10.3390/electronics13020361.
18. Slonopas A.. Zero Trust Cybersecurity and Why You Should Care about It, American Public University–Information-Technology-Blog. Available from:https://www.apu.apus.edu/area-of-study/information-technology/resources/zero-trust-cybersecurity-and-why-you-should-care-about-it.[Accessed 12 Dec. 2023].
19. Mehmood KT, Atiq S, Hussain MM. Enhancing QoS of Telecom Networks through Server Load Management in Software-Defined Networking (SDN). Sensors. 2023; 23(23):9324. https://doi.org/10.3390/s23239324.
20. Mehmood KT, Atiq S, Hussain MM, Sajid IA, Basiat MMA. Examining the Quality Metrics of a Communication Network with Distributed Software-Defined Networking Architecture. CMES. 2024; 141(2):1673-1708. https://doi.org/10.32604/cmes.2024.053903
21. Ming L, Leau YB, Xie Y. Distributed Denial of Service Attack in HTTP/2: Review on Security Issues and Future Challenges. IEEE Access 2024; vol. 12: pp. 33296-33308. Doi: 10.1109/ACCESS.2024.3371013.
22. Salmi S, Oughdir L, Affar AE. Defending Against App-Layer DDoS: Advanced Machine Learning Security.In 2024 4th International Conference on Innovative Research in Applied Science, Engineering and Technology (IRASET). 2024.FEZ, Morocco, pp. 1-6.doi: 10.1109/IRASET60544.2024.10548873.
23. Kathirkamanathan N, Thevarasa N, Mahadevan G, Skandhakumar N, Kuruwitaarachchi Prevention of DDoS Attacks Targeting Financial Services using Supervised Machine Learning and Stacked LSTM.In 2022 IEEE 7th International Conference for Convergence in Technology (I2CT). .2022. Mumbai, India, pp. 1-5.doi: 10.1109/I2CT54291.2022.9825228.
24. Leka E, Lamani L, Aliti A, Hoxha E. Web Application Firewall for Detecting and

Mitigation of Based DDoS Attacks Using Machine Learning and Blockchain. TEM Journal.2024; Vol 13(4): Pg 2802-2811.DOI: 10.18421/TEM134-17.
25. Paliwal S, Bharti V, Mishra AK. Machine learning combating DOS and DDOS attacks. International Journal of Business Information Systems (IJBIS).2022; Vol. 40(2):pp 177-191.https://doi.org/10.1504/IJBIS.2022.123638.
26. Sivakumar K, Thilagam SP. Vulnerability Testing of RESTful APIs Against Application Layer DDoS Attacks. International Journal of Advanced Computer Science and Applications(ijacsa).2025: Vol 16(3). http://dx.doi.org/10.14569/IJACSA.2025.01603110.
27. Dasari S, Kaluri R. An Effective Classification of DDoS Attacks in a Distributed Network by Adopting Hierarchical Machine Learning and Hyperparameter Optimization Techniques. IEEE Access 2024; vol. 12: pp. 10834-10845. doi: 10.1109/ACCESS.2024.3352281.
28. Islam U, Muhammad A, Mansoor R, Hossain MS, Ahmad I, Eldin ET, Khan JA, Rehman AU, Shafiq M. Detection of Distributed Denial of Service (DDoS) Attacks in IoT-Based Monitoring System of Banking Sector Using Machine Learning Models. Sustainability. 2022; 14(14):8374. https://doi.org/10.3390/su14148374.
29. Prashanth MV, Chethan MN, M Chinthan, Anushri R K, Lasya M. Distributed Denial of Services: Detection and Prevention. Research in Engineering and Management (IJSREM).2024; Vol 8(12): pg. 1-7. DOI: 10.55041/IJSREM39677.
30. Manaa EM, Hussain SM, Alasadi SA, Al-Khamees HAA. DDoS Attacks Detection based on Machine Learning Algorithms in IoT Environments. Inteligencia Artificial.2024; 27(74), 152–165. https://doi.org/10.4114/intartif.vol27iss74pp152-165
31. Saini PS, Behal S, Bhatia S. Detection of DDoS Attacks using Machine Learning Algorithms. In: 2020 7th International Conference on Computing for Sustainable Global Development (INDIACom), 2020, New Delhi, India, pp. 16-21. Doi: 10.23919/INDIACom49435.2020.9083716.
32. Aslam M et al. Adaptive Machine Learning Based Distributed Denial-of-Services Attacks Detection and Mitigation System for SDN-Enabled IoT.Sensors.2022; vol 22(2697). https://doi.org/10.3390/s2207269.
33. Djenna A, Saidouni DE, Abada W. A Pragmatic Cybersecurity Strategies for Combating IoT-Cyberattacks.In:2020 International Symposium on Networks, Computers and Communications (ISNCC). 2020. Montreal, QC, Canada, pp. 1-6, Doi: 10.1109/ISNCC49221.2020.9297251.
34. Aslan Ö, Aktuğ SS, Ozkan-Okay M, Yilmaz AA, Akin E. A Comprehensive Review of Cyber Security Vulnerabilities, Threats, Attacks, and Solutions. Electronics. 2023; 12(6):1333. https://doi.org/10.3390/electronics12061333.
35. Ali TE, Chong Y-W, Manickam S. Machine Learning Techniques to Detect a DDoS Attack in SDN: A Systematic Review. Applied Sciences. 2023; 13(5):3183. https://doi.org/10.3390/app13053183
36. Almaraz-Rivera JG, Perez-Diaz JA, Cantoral-Ceballos JA. Transport and Application Layer DDoS Attacks Detection to IoT Devices by Using Machine Learning and Deep Learning Models. Sensors. 2022; 22(9):3367. https://doi.org/10.3390/s22093367
37. Tharunika VS et al. Detection and Prevention of Advanced Persistent Threat (APT) Activities in Heterogeneous Networks using SIEM and Deep Learning.In 2023 14th International Conference on Computing, Communication and Networking Technologies (ICCCNT).2023. Delhi, India, pp. 1-8.doi: 10.1109/ICCCNT56998.2023.10306968.
38. Singh J, Behal S. Detection and mitigation of DDoS attacks in SDN: A comprehensive review, research challenges and future directions. Computer Science Review.2020; Vol.37(100279), https://doi.org/10.1016/j.cosrev.2020.100279.
39. Mishra N, Pandya S. Internet of Things Applications, Security Challenges, Attacks,

Intrusion Detection, and Future Visions: A Systematic Review. IEEE Access 2021; vol. 9: pp. 59353-59377. Doi: 10.1109/ACCESS.2021.3073408.

40. Hartono H, Wijaya RA, Khotimah K. Development of Detection and Mitigation of Advanced Persistent Threats Using Artificial Intelligence and Multi-Layer Security on Cloud Computing Infrastructure. International journal of artificial intelligence research.2024; vol 8(2). DOI: https://doi.org/10.29099/ijair.v8i2.1250
41. Yerriswamy T, Gururaj M. An Efficient Hybrid Protocol Framework for DDoS Attack Detection and Mitigation Using Evolutionary Technique.JTIT.2022; vol. 90(4): pp. 77–83. Doi: 10.26636/jtit.2022.165122.
42. Redekar R, Bharati R. DDoS Attacks: Detection Techniques, Challenges, and Modern Practices. In:2025 IEEE International Students' Conference on Electrical, Electronics, and Computer Science (SCEECS). .2025. Bhopal, India, pp. 1-7, Doi: 10.1109/SCEECS64059.2025.10940527.
43. Yungaicela-Naula NM, Vargas-Rosales C, Perez-Diaz JA. SDN-Based Architecture for Transport and Application Layer DDoS Attack Detection by Using Machine and Deep Learning. IEEE Access 2021; vol. 9: pp. 108495-108512. doi: 10.1109/ACCESS.2021.3101650.