GANShieldTrust: A Trustless Federated Learning Approach for Line-speed Low-rate DDoS Attack Protection in SDNs

Mohamed Lamine Adjou*, Chafika Benzaïd§, and Tarik Taleb¶
* University of Sciences and Technology Houari Boumediene, Algiers, Algeria
§ University of Oulu, Oulu, Finland
¶ Ruhr University Bochum, Bochum, Germany
Emails: madjou@usthb.dz, cbenzaid@acm.org, tarik.taleb@rub.de

***Abstract*—Low-rate Distributed Denial of Service (DDoS) attacks pose a serious threat in Software-Defined Networking (SDN) environments by degrading service quality while mimicking legitimate traffic. Existing Deep Learning–based defenses improve detection but remain vulnerable to dataset poisoning, rely on centralized controllers, and raise privacy and trust concerns. This paper introduces GANShieldTrust, a distributed and security-hardened framework that strengthens resilience against poisoning attacks by using GAN-generated adversarial training data to improve robustness against manipulated and unseen traffic. To remove single points of failure and support a zero-trust, decentralized architecture, feature extraction and classification are executed directly in SDN switches using P4-based Binarized Neural Networks (BNNs) for line-rate detection. To protect collaborative learning, GANShieldTrust integrates blockchain-secured federated learning, ensuring the integrity and authenticity of model updates while preventing malicious weight injection and preserving data privacy. The experimental results show 96.9% detection accuracy, strong resilience against poisoning and weight tampering, and low-latency mitigation, demonstrating its practicality for secure and scalable SDN deployments.**

***Index Terms*—SDN, Low-rate DDoS, GAN, Federated Learning, Blockchain, P4, and Zero-trust.**

I. Introduction

Distributed Denial of Service (DDoS) attacks remain a persistent threat in modern Software-Defined Networks (SDN), exploiting diverse techniques to exhaust shared resources such as bandwidth, memory, CPU, and server capacity. Low-rate DDoS attacks are particularly challenging because they target protocol-level congestion and session-control mechanisms while generating traffic that resembles legitimate users, often accounting for only 10–20% of total traffic [1]. Consequently, their impact may be misattributed to client or hardware issues, making detection and traceback difficult [2]. Existing detection methods range from statistical, spectral, and information-theoretic techniques [3] to deep learning (DL), which has shown strong effectiveness in identifying low-rate attacks by learning hidden flow characteristics and combining them with features such as inter-packet intervals and bandwidth [4]. However, DL models remain vulnerable to adversarial manipulation, in which attackers deliberately modify or inject inputs to mislead the model, while existing approaches for low-rate and unknown DDoS detection [5]–[12] often overlook dataset poisoning, centralized processing and its associated single points of failure, and privacy concerns arising from aggregated traffic data.

To address these limitations, this paper proposes *GANShieldTrust*, a resilient DL-based framework for detecting and mitigating low-rate DDoS attacks in SDN environments. The framework performs line-rate classification and mitigation directly at the switch level, reducing dependence on centralized processing and improving scalability and resilience. Secure Federated Learning preserves the privacy of local traffic data, while a lightweight blockchain-based protocol ensures the integrity of federated model updates through tamper-resistant verification and aggregation. This distributed architecture reduces single points of failure while strengthening robustness against adversarial attacks and compromised model updates.

The main contribution of this paper lies in systematically identifying and addressing multiple limitations of existing low-rate DDoS detection solutions within SDN environments, including centralized trust assumptions, slow and coarse-grained detection, and insufficient privacy guarantees. GANShieldTrust integrates and reinforces each of these aspects into a unified, SDN-compliant defense architecture. Specifically, the key contributions of GANShieldTrust are:

- Robustness against adversarial and unknown low-rate DDoS attacks through the use of GAN-generated synthetic datasets, which expose the detection model to diverse and deceptive traffic patterns beyond those found in conventional datasets.
- Decentralized and high-speed attack detection enabled by P4-implemented Binarized Neural Networks (BNNs) operating directly in the data plane, ensuring line-rate classification and per-flow mitigation without reliance on centralized inspection.
- Privacy-preserving and zero-trust federated learning, secured via a blockchain-based framework that guarantees integrity and authenticity of model updates while preventing weight poisoning and unauthorized manipulation.
- Practical effectiveness and scalability, demonstrated by achieving up to 96.9% detection accuracy with low latency and minimal computational and communication overhead, making the solution suitable for real-world SDN deployments.

Paper structure: Section II covers background concepts. Section III reviews related work. Section IV details the threat model. Section V explains the proposed solution and security analysis. Section VI outlines the implementation.

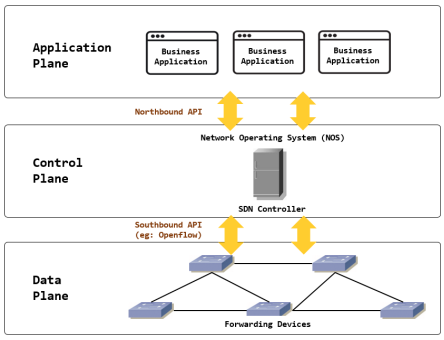


Fig. 1: SDN Overview.

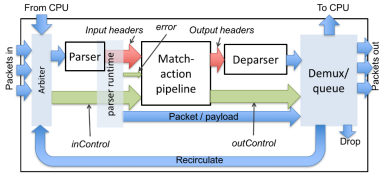


Fig. 2: The Very Simple Switch (VSS) Architecture from P4.

Section VII presents performance results. Section VIII concludes the paper.

II. Background Information

A. Software-Defined Networking

Software-Defined Networking (SDN) enables flexible and automated network management by decoupling the control and data planes. Applications define network policies through the application plane and communicate them to the centralized controller via northbound APIs. The controller uses real-time network information to install flow rules in forwarding devices through southbound APIs, while switches primarily handle packet forwarding using these flow tables (Fig. 1).

B. P4

Programming Protocol-independent Packet Processors (P4) [13] enhances the SDN data plane through vendor-independent pipeline programmability and high-speed packet processing (Fig. 2). It enables custom packet headers and programmable parsing, matching, and processing (Fig. 3). In this work, P4 performs flow-feature extraction and real-time traffic classification directly in the data plane for low-rate DDoS detection.

C. Low-rate DDoS Attacks

Attackers use DDoS attacks to disrupt services, cause financial or reputational losses, extort organizations, distract security teams, or gain competitive advantages. Such attacks commonly rely on botnets of compromised PCs, servers, and IoT devices that generate coordinated traffic toward a target [14]. DDoS attacks can be broadly classified into high-rate flooding attacks and low-rate attacks. Flooding attacks overwhelm targets with large traffic volumes, either through direct TCP, UDP, or ICMP flooding or through reflection/amplification techniques such as Smurf and DNS amplification. Although highly disruptive, their abnormal traffic volumes make them relatively easier to detect and typically require substantial attack resources.

In contrast, low-rate DDoS attacks deliberately use lower traffic rates and exploit protocol or application mechanisms rather than volume alone. HTTP Slowloris and

```
header ethernet_t {
    macAddr_t dstAddr;
    macAddr_t srcAddr;
    bit<16>   etherType;
}

header ipv4_t {
    bit<4>    version;
    bit<4>    ihl;
    bit<8>    diffserv;
    bit<16>   totalLen;
    bit<16>   identification;
    bit<3>    flags;
    bit<13>   fragOffset;
    bit<8>    ttl;
    bit<8>    protocol;
    bit<16>   hdrChecksum;
    ip4Addr_t srcAddr;
    ip4Addr_t dstAddr;
}
```

Fig. 3: P4 Headers Example.

slow GET/POST attacks maintain numerous connections by sending requests or fragments slowly, while Shrew attacks periodically generate short bursts that trigger TCP congestion control and reduce legitimate throughput. Other attacks exploit TCP reset/ACK behavior or manipulate congestion windows to degrade service with limited traffic. By mimicking legitimate traffic patterns, these techniques make detection and attribution more difficult [2], [3], [14]. Their lower bandwidth and potentially smaller botnet requirements can also reduce the attacker's operational cost while enabling prolonged attacks that evade conventional volume-based defenses.

Consequently, low-rate DDoS attacks can achieve persistent service degradation with relatively modest resources while remaining difficult to distinguish from legitimate traffic. This work therefore focuses on their detection and mitigation in SDN environments.

D. Deep Neural Networks and Generative Adversarial Networks

Deep Neural Networks (DNNs) employ multi-layer architectures for advanced feature extraction and classification. Deep learning (DL) includes supervised approaches, such as MLPs, CNNs, and RNNs, which learn from labeled data, and unsupervised methods, such as AutoEncoders and k-NN, which identify patterns in unlabeled data [15]. Although DL provides high accuracy for low-rate DDoS detection, it remains vulnerable to adversarial examples [2].

Generative Adversarial Networks (GANs), introduced by [16] in 2014, generate synthetic data that approximate real data distributions. A GAN consists of a generator G, which produces synthetic samples, and a discriminator D, which distinguishes them from real samples. Through adversarial min-max training, G progressively improves the quality of generated data. Although widely used in computer vision [17], GANs have also been applied to generate adversarial examples for DNN training, including low-rate DDoS detection [8], [18]. Accordingly, this work employs GAN-based data generation to improve the robustness of the DNN model against adversarial attacks.

E. Federated Learning

Traditional machine learning typically requires centralized data collection and powerful servers, raising scalability and privacy concerns, particularly with the growth of IoT and mobile devices. Federated Learning (FL) addresses these challenges by enabling local nodes to train models

on their own data and share only model parameters with a central server for aggregation and redistribution [19]. This preserves data privacy while supporting distributed training. In this work, FL trains local classifiers from globally aggregated parameters, with model updates secured through a lightweight blockchain-based protocol.

F. Blockchain

Originally introduced as the foundation of Bitcoin [20], blockchain enables secure, decentralized transactions without requiring a trusted central entity. It achieves this by recording digitally signed transactions in a chain of blocks, each containing a hash of the previous block and a miner-generated value. Once validated, new blocks are broadcast and appended across all participating nodes.

In this work, a custom blockchain-based protocol secures the weight exchange in the federated learning process, ensuring integrity and tamper resistance.

III. Related Work

Low-rate and application-layer DDoS attacks have attracted growing academic attention, prompting the development of various detection and mitigation techniques. This section presents the main DL-based approaches targeting these attacks.

In [5], researchers proposed a binary classification method using DNNs and LSTM, trained on the CICIDS2017 dataset and tested on a simulated dataset (ANTS2019) generated by replaying CICIDS2017 traffic in Mininet. Similarly, [6] introduced a hybrid model combining 1D-CNN, GRU, and DNN to detect both flooding-based and low-rate DDoS in SDNs, using datasets from CIC-DDoS2019 and traffic generated with tools like Scapy and Hping3. Both solutions achieved over 99.8% accuracy. However, these approaches rely on datasets generated with the same tools, limiting generalization to unseen attacks. Additionally, deep neural networks (DNNs) incur high processing time for packet classification. To mitigate this, we use a binarized neural network (BNN) for faster inference, and a GAN model to generate more diverse training samples evaluated on real-world traffic.

In [7], the authors use BNNs with Federated Learning to detect DDoS attacks. BNN classifiers are deployed on P4-programmed switches for fast packet classification, while real-valued classifiers run on edge controllers. A cloud server aggregates weights from these controllers. This architecture delivers strong performance on BOTNET2014 and CICIDS2017, comparable to NN, DT, and SVM classifiers. However, it remains vulnerable to adversarial and poisoning attacks and relies on a three-tier architecture that increases attack surfaces and single points of failure. In contrast, our solution adopts a fully distributed model and secures weight exchange through blockchain.

In [10], a machine learning-based method addresses low-rate DDoS by training an ANN model to detect traffic anomalies linked to TCP congestion manipulation. Suspicious flows are blocked if they resemble learned benign patterns. The system employs Weighted Federated Learning, where clients assign model weights based on local accuracy, leading to a robust global model. With 95.24% detection accuracy, the solution offers adaptability and strong SDN protection without degrading performance.

In [8], the authors propose a defense against various DDoS attacks using adversarial training in an SDN environment. A GAN model is used alongside a *Safe List* of trusted IPs to distinguish adversarial DDoS from legitimate traffic.

Similarly, [9] presents an SDN module that uses a DNN trained against white-box FGSM attacks to detect application-layer DDoS. The training involves perturbing input data with a one-step gradient method. The model achieves 99.65% accuracy on the CICIDS2017 dataset.

Despite their effectiveness, these approaches ([8]–[10]) share several limitations. They rely heavily on centralized processing at the SDN controller, which introduces latency and potential security risks. Moreover, they assume reliable conditions during training (e.g., honest clients or specific adversarial settings), limiting robustness against adversarial manipulation. Finally, their focus on specific attack types or constrained datasets reduces their ability to generalize to diverse and unseen threats.

In [11], the authors employ an online learning approach using the Passive-Aggressive (PA) classifier, which updates its weights and bias incrementally with each data instance. Misclassifications trigger aggressive updates, with the learning rate determined by the loss and regularization terms. This enables continuous adaptation to evolving data and improves performance over time. Compared to batch learning, which retrains on full datasets, PA offers faster and more efficient model updates, achieving 99.7% accuracy. While semi-supervised or transfer learning could also be considered, they differ in design and applicability.

In [12], GASF-IPP is proposed to detect and mitigate low-rate DDoS attacks within the SDN data plane. It analyzes traffic and IP-port data using the OpenFlow protocol and applies Gramian Angular Summation Field (GASF) transformation to time-series traffic features for anomaly detection. Identified attackers and victims are then targeted with specific flow rules. The framework demonstrates real-time effectiveness, with low false positive rates and an average response time of 6.77 seconds.

GANShieldTrust addresses each identified limitation, with every component of the solution thoroughly detailed, evaluated, and its results systematically compared against existing approaches within the remainder of the paper.

To the best of our knowledge, no existing solution jointly addresses privacy and weight poisoning in a unified, robust framework.

Table I provides a comparative overview of key DL-based approaches, highlighting their techniques, datasets, advantages, and limitations.

IV. Threat Model

We consider an attacker launching low-rate DDoS attacks against application-layer services (as defined in Section II-C). The attacker is assumed to possess full knowledge of the network topology, the software stack of target servers, and the ML-based detection models in use.

In zero-trust network security models, compromise of a single node does not imply full control over the infrastructure. Attackers often seek persistence and stealthiness rather than immediate disruption, especially in monitored SDN environments. Low-rate DDoS attacks remain effective in this context because they mimic legitimate traffic, degrade service quality gradually, and evade threshold-based

TABLE I: Comparison of DL-based solutions for low-rate and application-layer DDoS attacks.

Solution	Key Technique	Dataset Used	Advantages	Limitations
[5]	Binary classification with DNN and LSTM	CICIDS2017, ANTS2019	High accuracy (99.8% and 99.9% for DNN and LSTM respectively); effective for simulated datasets.	Limited generalization to unknown attacks; relies on similar training/testing datasets.
[6]	Combination of 1D CNN, GRU, and DNN frameworks	CIC-DDoS2019, generated datasets with Scapy and Hping3	High accuracy (99.81% and 99.88%, from both datasets respectively); capable of detecting low-rate DDoS.	Similar tools for generating training and testing data limit generalization.
[7]	BNNs implemented in P4 with federated learning at the edge	BOTNET2014, CICIDS2017	Enables line-speed detection; privacy-preserving with federated learning.	Vulnerable to adversarial and weight poisoning attacks; introduces a three-tier architecture.
[10]	ANN with Weighted Federated Learning	Real-world and synthetic datasets.	Adaptive Weighted Federated Learning for global model updates; effective low-rate DDoS detection (95.24%).	Relatively lower detection accuracy compared to other methods that use ANN.
[8]	GAN with adversarial training and a Safe List	Emulated data, CICDDoS2019	Detects adversarial examples; utilizes SDN controller for flow collection.	High latency due to centralized feature processing; privacy concerns from flow centralization.
[9]	DNN model resilient to FGSM adversarial attacks	CICIDS2017	High accuracy (99.65%); defenses against white-box adversarial attacks	Limited adversarial training (FGSM only); centralized processing at the SDN controller
[11]	Passive-Aggressive (PA) classifier for online learning	Incremental datasets	Adapts continuously to new data; efficient for streaming datasets; high accuracy (99.7%).	May require fine-tuning for initial deployment.
[12]	GASF-IPP with timing analysis and IP-port data	Real-time SDN traffic	Accurate real-time detection; low false positive rate; fast response time (6.77s)	Limited scalability for large networks.
GANShieldTrust	Distributed detection with GAN-generated datasets and blockchain-based federated learning	GAN-generated dataset, Real-world validated attacks	Distributed processing; secure and private federated learning; strong resilience to adversarial examples; scalability for large networks.	May require additional resources for local training.

alarms [21]. Launching such attacks from a compromised node enables insider-like behavior that bypasses perimeter defenses while preserving attacker access.

Within the scope of low-rate DDoS attacks considered in this work, the attacker's capabilities are defined as follows:

- **Low-rate DDoS attack:** Initiating attacks from either inside or outside the network, possibly from multiple distributed sources targeting different services simultaneously, as low-rate DDoS attacks are stealthy, mimic legitimate traffic behavior, and therefore constitute the primary threat focus of this study.
- **Flow injection:** Introducing crafted flows during training to perform adversarial attacks that mislead the model into misclassifying low-rate DDoS traffic [22], as ML-based detection systems are inherently vulnerable to data poisoning that can bias the learned decision boundary.
- **Weight interception:** Intercepting and altering weight updates in the federated learning process, enabling poisoning attacks that reduce model accuracy or backdoor attacks that insert hidden triggers into model parameters, since federated learning expands the attack surface to the model exchange channel where integrity violations may occur.
- **Hijack network nodes:** Gaining control over switches or the SDN controller to inject falsified weights or allow malicious traffic to bypass detection mechanisms, reflecting realistic risks in software-defined networks where programmable data-plane devices and centralized controllers can become compromise targets.

To ensure reliable detection and secure federated aggregation, GANShieldTrust assumes that a strict majority of participating nodes remain honest and operational. This follows established principles in Byzantine-resilient distributed systems, where majority agreement supports reliable validation. Unlike classical Byzantine Fault Tolerance (BFT) protocols, which typically tolerate up to one-third malicious nodes due to multi-phase consensus requirements, GANShieldTrust uses majority-based validation $(N/2+1)$ for weight verification and attack confirmation. Since all nodes have equal roles and voting power, without designated leaders or weighted influence, a simple majority is sufficient to prevent malicious nodes from overriding honest decisions. Thus, if fewer than half of the nodes are compromised, forged weights or false detection reports cannot obtain majority approval.

In practical SDN deployments, network segmentation, least-privilege access control, continuous monitoring, firmware validation, and hardware trust anchors can further reduce the risk of majority compromise. Controlling more than half of the switches would constitute a systemic breach

Algorithm 1 Switch Algorithm for Attack Detection.

Require: Flow features F
Ensure: Traffic forwarding decision and dataset update
$attack_type \leftarrow$ BNN_Classification(F)
if $attack_type =$ *low-rate DDoS* **then**
  Install drop rule for flow F in flow table
  Send F to controller for verification
  log: Attack detected and traffic blocked
  $entry \leftarrow$ Receive confirmation from controller
  Add $entry$ to local dataset
else
  Forward flow normally
  log: benign detected
end if

Algorithm 2 Controller Algorithm for Attack Detection.

Require: Reported attack set R for flow F, set of online switches S
Ensure: Network-wide confirmation decision
$n \leftarrow |R|$
$threshold \leftarrow |S|/2$
if $n \geq threshold$ **and** all reports classify F as *low-rate DDoS* **then**
  for all switch $s \in S$ **do**
    Send confirmation (F, True) to s
  end for
  log: Confirmed low-rate DDoS attack
else
  for all switch $s \in S$ **do**
    Send confirmation (F, False) to s
  end for
  log: Insufficient evidence, continue monitoring
end if

of the administrative domain and is therefore considered beyond the protection scope of GANShieldTrust.

Thus, the honest-majority assumption is a justified condition for the proposed majority-based distributed integrity mechanism.

V. Proposed Solution

A. Overview

This paper introduces GANShieldTrust, a robust solution for mitigating low-rate DDoS attacks in SDN networks, as depicted in Fig. 4. The design is guided by four core principles:

1. **Zero-trust:** No central trusted entity is assumed. Attack detection and mitigation are fully distributed and supported by a trustless verification mechanism.
2. **Adaptive security:** The use of GAN-generated datasets strengthens local classifiers against adversarial inputs, while federated learning reduces the impact of dataset poisoning.
3. **Speed:** Traffic features are extracted and classified at line-rate within the data plane.
4. **Privacy:** Neither traffic nor flow features are shared outside the local switch, preserving data confidentiality.

The DNN is implemented directly in the data plane using P4, enabling each switch to detect attacks and classify traffic at line rate. To address P4's limitations and speed up detection, we adopt Binarized Neural Networks (BNNs) [23], which use binary weights and activations while maintaining accuracy comparable to conventional models [7], [24]. Switches locally extract features, perform

Algorithm 3 Switch Algorithm for Secure Federated Learning.

Require: New dataset entries D, threshold n, set of online switches S
Ensure: Verified global model update
if $|D| \geq n$ **then**
  Train local BNN model
  $W \leftarrow$ extract local model weights
  Send W to Controller
  $h_w \leftarrow \text{hash}(W)$
  Broadcast h_w over blockchain overlay
end if
if node is elected miner **then**
  Collect hashes $\{h_1, h_2, \dots\}$ from switches
  if all switch hashes received **then**
    Receive aggregated hash h_{agg} from Controller
    $h_b \leftarrow \text{hash}(\{h_1, h_2, \dots\} \cup h_{agg})$
    Broadcast block hash h_b
  end if
else
  Receive aggregated weights W_{agg} from Controller
  Receive controller block hash h_c
  Receive miner block hash h_m
end if
if $h_c == h_m$ **then**
  Accept update and set model weights to W_{agg}
  log: Secure federated learning successful
else
  Report mismatch to Controller
  log: Integrity verification failed
end if

Algorithm 4 Controller Algorithm for Secure Federated Learning.

Require: Set of online switches S
Ensure: Verified aggregated global model weights
Initialize weight list $W \leftarrow \emptyset$
Initialize hash list $H \leftarrow \emptyset$
for each switch $s \in S$ **do**
  Receive local weights W_s from switch s
  $W \leftarrow W \cup \{W_s\}$
end for
for each switch $s \in S$ **do**
  Receive weight hash h_s from blockchain overlay
  $H \leftarrow H \cup \{h_s\}$
end for
$W_{agg} \leftarrow$ compute average of W
$h_{agg} \leftarrow \text{hash}(W_{agg})$
$h_{block} \leftarrow \text{hash}(H \cup h_{agg})$
Broadcast h_{block} via blockchain overlay
for each switch $s \in S$ **do**
  Send aggregated weights W_{agg} to switch s
end for

classification, and train the BNN with adversarial correction (Algorithms 1 and 3). The updated weights are sent to the SDN controller, which averages and returns them (Algorithm 2 and 4). This process accelerates learning, improves robustness, and preserves traffic privacy. To ensure the integrity of the weights exchanged, a lightweight blockchain protocol [25] broadcasts signed weight hashes for distributed verification.

Fig. 4 illustrates GANShieldTrust's operation when a low-rate DDoS attack targets a server hosting application services. The main steps are the following:

1. **Low-rate DDoS attack:** An attacker, internal or external, sends traffic to exhaust application resources and deny service access.

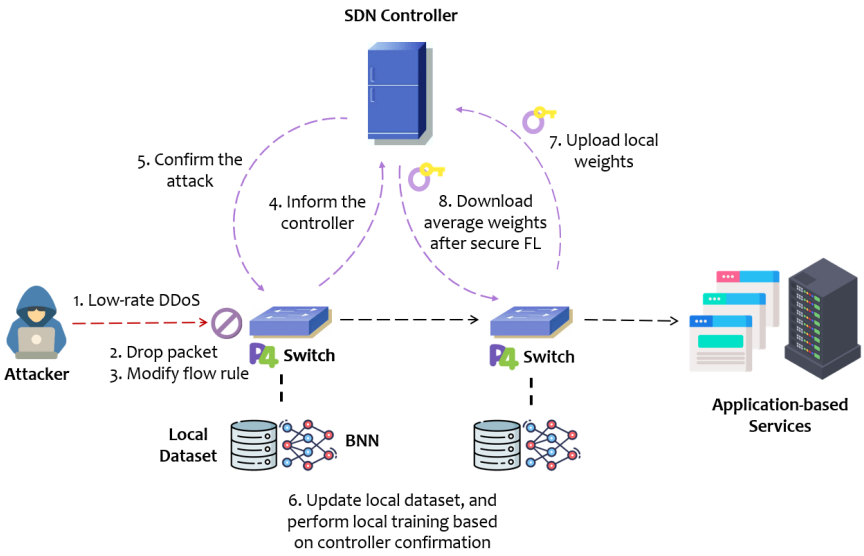


Fig. 4: Overview of GANShieldTrust Solution.

2. **and 3) P4 switch attack detection:** Upon receiving traffic, a P4 switch extracts flow features and classifies them using its locally trained BNN model (see Section VI-A). If classified as 'low-rate ddos', the switch blocks the traffic via flow table updates. In addition to blocking, switches can apply rate limiting to mitigate impact or reroute suspect flows through an IDS for deeper inspection while preserving availability.
4. **and 5) Consensus-based attack confirmation:** Upon detecting an attack, the P4 switch sends flow feature values to the SDN controller. The controller verifies if other switches on the same path report the same flow. If a majority confirms, the controller validates the attack and instructs the switches to label the flow as 'low-rate ddos' or 'benign' accordingly.
6. **7) and 8) Switch model update with secure federated learning:** Once enough labeled entries are collected (details in Section VI-D for more details), each switch retrains its local BNN using real-valued weights. Rather than fixed-size retraining, updates are triggered by performance drops (e.g., $\geq$ 5% accuracy loss) or anomalies such as switch disagreement or novel traffic. This ensures timely adaptation to evolving threats. After training, the controller performs secure federated learning by averaging weights and redistributing them to switches (details in Section VI-E).

Algorithms 1–4 detail the methods employed for Attack Detection and Secure Federated Learning.

B. Security Analysis

We consider an attacker launching low-rate DDoS attacks in an SDN network secured by GANShieldTrust. The detection and mitigation model includes:

- **Protected assets:** Application services and servers exposed to internal or external users.
- **Attacker:** Familiar with the network topology, capable of launching distributed low-rate DDoS, intercepting weight exchanges, injecting crafted flows, and spoofing node identities.
- **Defense mechanism:** A BNN-based model deployed in each switch, trained on GAN-generated data to detect evasive and unknown low-rate DDoS. Secure Federated Learning aggregates model weights to ensure robustness, privacy, and protection against poisoning (details in Section VI-E).

  In this work, we assume that adversarial methods can be leveraged to make low-rate DDoS attacks stealthier, and thus harder to detect by ML-based detectors. Our use of GANs serves to train more robust classifiers that can recognize such evasive, adversarially-shaped traffic.

This distributed approach offers a resilient defense without requiring additional infrastructure or SDN protocol changes.

In the following, various attack vectors and their mitigation methods are detailed:

1) Attacker launches low-rate DDoS attacks: Regardless of the attack origin, GANShieldTrust enables each participating switch to detect and mitigate low-rate DDoS traffic locally using its BNN classifier. The GAN-trained BNN is implemented in the P4 data plane, enabling line-rate classification and rapid mitigation close to the traffic source. Although misclassification remains possible, distributed deployment limits its impact and supports low-latency mitigation. The evaluation in Section VII demonstrates high detection accuracy, low false-positive rates, and effective generalization to unseen attack variants.

2) Attacker injects fabricated network flows: An attacker may inject fabricated flows to bias local BNN training. GAN-based data augmentation improves the diversity and robustness of the training data, thereby reducing the impact of distribution shifts and adversarially manipulated flows, as evaluated in Section VII.

3) Attacker intercepts/modifies BNN weights: To protect federated weight updates, GANShieldTrust incorporates a secure federated learning mechanism inspired by *TopoTrust* [25]. A lightweight blockchain overlay verifies the integrity of exchanged weights using hash-based validation and consensus. Tampered updates are detected before aggregation, preserving the integrity of the global model with limited communication and computational overhead.

4) Attacker takes control over one or more network nodes: If a node is compromised and attempts to inject falsified weights, hash inconsistencies are detected by

honest peers during blockchain verification. The distributed deployment also limits the impact of individual node failures or compromises and maintains detection availability under multi-source attacks. The resilience of the proposed architecture is evaluated quantitatively in Section VII.

VI. Solution Implementation

The proposed solution is deployed as a lightweight module within SDN-enabled switches and the SDN controller, requiring no changes to the existing network topology. Although offloading intelligent functions to the data plane may appear contrary to SDN's centralized paradigm, it reduces the scalability, privacy, and security risks associated with centralized detection and mitigation [26]. Control remains centralized at the SDN controller, while selected security functions are distributed to the data plane, consistent with the zero-trust model [27] and leveraging modern high-performance switching capabilities.

In the following, we detail the implementation of each component of the solution illustrated in Fig. 5.

A. GAN-generated Low-rate DDoS Training Dataset

In this work, various low-rate DDoS datasets have been used as basis to train two types of GAN for tabular data; i.e., the flow features. The first is for the traffic labeled as 'benign' and the other for the one labeled as 'low-rate ddos', which encompasses several types of low-rate DDoS attacks. The GAN model is then used to generate two corresponding datasets for 'benign' and 'low-rate DDoS' to train the BNN model.

The datasets used for GAN training are the following:

- **CIC DoS dataset (2017)** [28]: The attacks were generated by the authors using a testbed environment with a target of ten web servers running Apache Linux v.2.2.22, PHP5 and Drupal v.7 as a content management system. Three high-volume HTTP attacks were launched: DoS improved GET (Goldeneye), DDoS GET (ddossim), and DoS GET (hulk). In addition, five low-volume HTTP attacks were deployed: slow-send body (Slowhttptest), slow-send body (RUDY), slow-send headers (Slowhttptest), slow-send headers (Slowloris), and slow-read (Slowhttptest).
- **Kaggle's Low Rate DDoS Attack Dataset** We also incorporated traffic traces from Kaggle's Low Rate DDoS Attack Dataset [29], which consists of raw network captures (PCAP files, notably an ICMP.pcap of approximately 2.41 GB) containing network traffic relevant to low-rate DDoS scenarios. Although detailed labeling or processed flow features are not specified in the public listing, we utilized these traces to enrich our training data, particularly enhancing coverage of ICMP-based low-rate DDoS behaviors, and augmented them into flow-level datasets in line with our feature extraction framework.
- **CSE-CIC-IDS2018 on AWS** [30]: The dataset is built using the network captures and system logs of seven different attack scenarios, namely: Brute-force, Heartbleed, Botnet, DoS, DDoS, Web attacks, and infiltration of the network from inside.

We extracted the selected flow features listed in Table II and normalized them to construct a custom dataset from multiple sources. The dataset comprises 50% of the CIC DoS Dataset (2017), selected for its high proportion of low-rate DDoS flows, 35% of the Kaggle Low Rate DDoS Attack Dataset, and 15% of CSE-CIC-IDS2018 on AWS. The resulting dataset contains 150,000 entries and is periodically enriched with newly observed and switch-level classified flows, whose labels are validated through controller consensus (Section VI-C). Only flow-level feature summaries and validated labels are retained; raw packets and sensitive traffic traces remain local to each switch.

This design is consistent with federated learning principles, as no raw traffic data are transmitted outside the network fabric. Instead, switches contribute validated knowledge, such as labels, model updates, or synthetic representations, enabling the dataset to evolve from distributed observations while preserving data privacy.

Both the Generator and Discriminator use 256 neurons per layer. This configuration was empirically selected to provide sufficient capacity for modeling complex flow-level correlations without introducing significant overfitting or training instability. Smaller configurations (64–128 neurons) resulted in underfitting and lower sample diversity, whereas larger architectures provided no measurable accuracy improvement while increasing training instability.

The remaining hyperparameters were similarly determined through empirical evaluation. Area Under the ROC Curve (AUC) was selected as the adversarial evaluation metric because it provides a more reliable measure of class separability for imbalanced low-rate DDoS data than accuracy alone. The Adam optimizer with a weight decay of $1e{-}6$ was adopted to improve regularization and convergence stability. A batch size of 500 produced smoother gradient updates and reduced discriminator oscillations. Finally, a learning rate of 0.02 provided the best trade-off between convergence speed and synthetic data quality; lower values slowed convergence, whereas higher values increased the risk of mode collapse.

In GANShieldTrust, the resource-intensive GAN training is delegated to the SDN controller, which possesses greater computational capacity, while preserving the system's trustless architecture. The controller generates adversarial datasets and updates the global model without accessing raw traffic data, adhering to federated learning principles that keep sensitive information local to each switch. Blockchain integration further secures model update exchanges, ensuring tamper resistance and decentralized verification. This trustless design eliminates reliance on any single trusted entity, maintaining the integrity and robustness of the detection process throughout the SDN infrastructure.

To quickly retrain the GAN model in GANShieldTrust with newly detected attacks, transfer learning is applied by freezing earlier layers of the GAN and fine-tuning only the later layers with the new data. This minimizes computational cost while effectively integrating novel attack patterns. Adaptive data augmentation is also used to create diverse variants of the new attacks, enhancing the model's generalization to similar future threats.

To support continual learning, dynamic replay buffers maintain a balanced mix of past and recent attack samples for robust fine-tuning. In a federated setting, switches share local representations of new threats, contributing to a more comprehensive global GAN update. This ensures the model evolves with changing traffic patterns across the network.

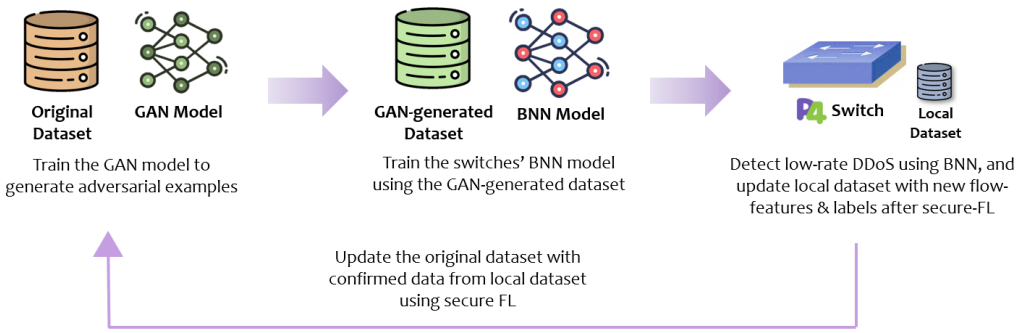


Fig. 5: Model Training and Dataset Update.

While secure federated learning ensures model integrity, advanced poisoning strategies such as slow-drift attacks remain an active research area; future enhancements will incorporate update anomaly detection and trust-weighted aggregation to mitigate gradual model biasing.

B. Flow Features Extraction

In this solution, flow features are selected experimentally to optimize the trade-off between extraction overhead and detection accuracy. These features are computed in P4 within a sliding time window δ, initially set to 2 $seconds$. When a suspected attack causes increased server response latency, the switch adapts by extending δ to twice the maximum observed latency within the same flow. This dynamic adjustment enables the capture of more relevant flow characteristics for improved detection.

The flow features selected for this solution are shown in Table II. Concretely, after δ seconds, the P4 program verifies whether at least 10 packets from the same flow have traversed the switch. If this condition is not met, δ is increased to allow further packet collection before triggering BNN classification.

TABLE II: Flow features

Flow Feature	Description
$Tot_Lgth_Fwd_Pkt$	The value in bytes of all packets with the <IP, destination port> of the target server during δ.
$Flow_IAT_Min$	The minimum value in milliseconds of inter-packet time interval during δ.
$Flow_IAT_Max$	The maximum value in milliseconds of inter-packet time interval during δ.
Fwd_IAT_Tot	The total value in milliseconds of inter-packet time interval during δ.
$Flow_Dur$	The duration of the session to the server in milliseconds
$Fwd_Pkt_Lgth_Min$	The minimum length in bytes of the packets targeting the server during δ.
$Fwd_Pkt_Lgth_Max$	The minimum length in bytes of the packets targeting the server during δ.
Tot_Fwd_Pkt	The number of packets targeting the server during δ.

C. Traffic Classification

Once flow features are extracted, they are passed to the BNN model, which outputs a binary classification: 'low-rate-ddos' or benign'. The BNN is implemented in P4 with an input layer, a 1024-neuron hidden layer, and an output layer, using only bitwise operations due to $P4_{16}$ constraints. This design follows the approach first introduced in [7].

By default, all flows are permitted. If a switch detects an attack, it updates its flow table to block the traffic and notifies the controller. The controller cross-checks this with reports from other switches along the same path. If a majority confirm the attack, it is validated and recorded as 'low-rate ddos' in their local datasets. Otherwise, it is marked as 'benign'.

D. BNN Training

Each switch performs local BNN training using real-valued weights and minimizes a binary cross-entropy loss via stochastic gradient descent. The required training-set size was determined using learning curves, progressively increasing the dataset from 20% to 100% while monitoring detection accuracy. Results showed diminishing accuracy gains beyond approximately 10,000 validated traffic entries, indicating a stable learning region. The final training set therefore contains 10,000 validated and balanced observations of low-rate DDoS and legitimate traffic.

The trained weights are periodically transmitted to the SDN controller through the secure federated learning process (Section VI-E), where they are integrity-verified and aggregated using weighted averaging before redistribution. Switches then update their local models and binarize the weights into bit strings for deployment in the P4 pipeline, enabling line-rate inference.

This approach combines the precision of real-valued weights during training with the computational and memory efficiency of binary weights during inference. The BNN is implemented using Larq[1], an open-source library based on TensorFlow and Keras for developing binarized neural networks.

1) Feasibility of Local DNN Training in SDN via SmartNICs and DPUs: A key challenge in distributed SDN

[1]https://larq.dev/

security is the feasibility of local deep learning training. In our architecture, each node trains a single-layer DNN with 1024 hidden neurons, after which the weights are binarized for P4-based line-rate inference. While conventional ASICs cannot support local training, programmable SmartNICs and Data Processing Units (DPUs) provide sufficient computing resources for such workloads. Platforms such as NVIDIA BlueField-2/3 and AMD Pensando support packet processing, analytics, and machine learning, including federated learning [31]. Commercial switches, including HPE Aruba CX 10000 and Cisco Catalyst 8102-DPU, further demonstrate the integration of DPUs into programmable networking platforms [32], [33].

In our implementation, Larq-based training runs asynchronously on SmartNIC/DPU compute resources, outside the P4 forwarding pipeline. After training, weights are binarized and deployed in P4 for efficient bitwise inference. This separation preserves line-rate forwarding while enabling scalable and privacy-preserving local learning.

E. Secure Federated Learning

After local BNN training (performed offline), each switch sends its computed weights to the controller, which calculates the average. To ensure authenticity and integrity of this weight exchange, GANShieldTrust adopts a mechanism inspired by *TopoTrust* [25], leveraging a lightweight proof-of-selection scheme rather than computationally expensive consensus (e.g., Proof of Work). In this process, the miner responsible for hashing is randomly selected by all nodes using the previous block's hash, making targeted attacks impractical.

Each switch broadcasts a 256-bit hash of its weight vector to all other nodes over the blockchain overlay network concurrently with transmitting the actual weights to the controller. Once the controller computes the average weights, it also broadcasts the hash of this aggregate on the same overlay (Fig. 6). The elected miner collects all switch hashes and the controller's aggregated hash, forms a new block, and broadcasts its hash to all nodes.

Due to the lightweight design:

- Block creation time is typically below 5–10 ms, as it only involves hash aggregation and block formatting.
- Consensus delay (hash broadcast and miner selection) remains under 10–20 ms, since no iterative validation or cryptographic puzzle solving is required.
- Verification cost per node is negligible (hash recomputation and comparison), typically $<$1 ms CPU time, introducing less than 1% computational overhead during federated learning rounds.

Local verification mismatches trigger alerts indicating potential attacks. The Secure Federated Learning mechanism introduces limited overhead: less than 30 ms latency per round, approximately 1.2% bandwidth overhead from exchanging 256-bit hashes, and less than 1% CPU overhead for hashing. Thus, blockchain-based integrity protection preserves real-time performance while providing tamper resistance. To mitigate non-independent and identically distributed (non-IID) data, where switches observe heterogeneous and statistically different traffic distributions, GANShieldTrust combines clustered and personalized federated learning with GAN-based augmentation and weighted aggregation. Switches with similar traffic distributions are grouped during aggregation, while local fine-tuning adapts the global model to node-specific characteristics. GAN-generated samples compensate for rare low-rate DDoS patterns, and weighted aggregation prevents poorly representative datasets from disproportionately influencing the global model. Adaptive learning rates and early stopping further improve convergence stability. Clustering is performed dynamically by the controller and does not affect real-time packet classification or mitigation latency. Experimental results in Section VII-H confirm stable convergence and improved detection accuracy. For resource efficiency, switches retain only the latest blockchain block hash rather than the complete chain. If the elected miner becomes unavailable, a new miner is selected after a 4-second timeout using the same hash-based mechanism, ensuring continuity with minimal storage and coordination overhead. In GANShieldTrust, incorporating blockchain into the secure federated learning process provides significant benefits but also introduces certain trade-offs that must be carefully managed to achieve optimal performance, as outlined below:

1) Utility and Benefits: Blockchain enhances the integrity and security of federated learning by ensuring that SDN switch weight updates are authentic and tamper-resistant. Its decentralized trust model helps mitigate data poisoning and model tampering by enabling detection of anomalous or adversarial updates. This supports a *zero-trust security model* while providing a tamper-proof audit trail that strengthens accountability and privacy protection for sensitive flow data in distributed SDN environments.

2) Drawbacks and Challenges: However, blockchain integration introduces computational and communication overhead, particularly for consensus and block verification, which may delay federated model updates and reduce real-time responsiveness. As the number of participating nodes increases, blockchain storage and bandwidth requirements also grow, potentially limiting scalability in resource-constrained or latency-sensitive SDN environments.

3) Balancing Utility and Drawbacks: To reduce blockchain overhead, GANShieldTrust employs lightweight consensus and pruning to retain only essential recent data, while blockchain is applied selectively to critical model updates. Differential Privacy (DP) adds controlled noise to shared detection results, limiting the reconstruction of sensitive traffic patterns with negligible overhead (typically below 1% in federated settings) [34]. Federated aggregation further preserves confidentiality by exchanging only global model statistics.

These mechanisms follow a resource-aware design: training is asynchronous and offline, while only model weights and small cryptographic hashes are exchanged, limiting bandwidth overhead to approximately 1–2%. The blockchain stores hashes rather than raw data and uses lightweight consensus to reduce validation costs. Together, these techniques protect model integrity, mitigate poisoning, and preserve privacy while minimizing the impact on forwarding and real-time detection performance.

VII. Solution Evaluation

This section evaluates both the protection effectiveness and latency overhead of the proposed solution. Each component is assessed individually by comparing system behavior with the component enabled versus disabled,

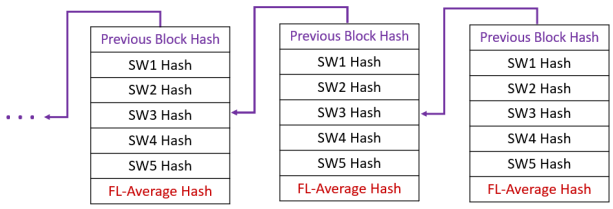


Fig. 6: Secure Federated Learning Blockchain Example.

TABLE III: Confusion Matrix

		Predicted classification	
		'lr-ddos'	'benign'
Real Classification	'lr-ddos'	True Positives (TP)	False Negatives (FN)
	'benign'	False Positives (FP)	True Negatives (TN)

while keeping the rest of the system active. Additionally, the overall detection performance is benchmarked against alternative detection algorithms.

The evaluation covers the following scenarios:

1) **BNN Classification**: Compared with top machine learning algorithms: DT, SVM, RF, and real-value DNN for low-rate DDoS detection.
2) **GAN-based Training**: Assessed against BNNs trained on original datasets to detect both known and unknown attacks.
3) **Federated Learning**: Compared with isolated single-node classification.
4) **Secure Federated Learning evaluation**: Tested against weight interception via man-in-the-middle attacks.
5) **Overall performance**: Benchmarked against other low-rate DDoS defense solutions.
6) **Mitigation latency**: Measured by server response delays during and after attacks.
7) **Impact of** δ: Evaluated for detection accuracy and time under varying δ values.
8) **Multi-source Attacks**: Tested with attacks from multiple origins to assess detection.
9) **Overhead evaluation**: Measured for computational and communication costs from GANShieldTrust and secure federated learning.

For consistency, each scenario is repeated. Averages are used if results vary slightly; otherwise, parameters are adjusted until stable outcomes are achieved.

A. Evaluation Metrics

Standard metrics for DNN evaluation have been used, including Accuracy (AC), which measures the proportion of correctly classified samples among all samples; Precision (PR), which measures the proportion of correctly predicted positive samples among all samples predicted as positive; Recall (RC), which measures the proportion of correctly identified positive samples among all actual positive samples; and F-measure (F1), which represents the harmonic mean of Precision and Recall and provides a balanced measure of their performance. The AC measures are taken from the True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN), derived from the confusion matrix in Table III. PR is the fraction of correctly identified attacks among all attacks identified by the model, while RC calculates this fraction among all attacks present in the dataset. Moreover, F1 combines PR and RC to calculate their harmonic mean, reflecting their values into a single one. The metrics are calculated as follows:

$$Accuracy\,(AC) = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$Precision\,(PR) = \frac{TP}{TP + FP} \tag{2}$$

$$Recall\,(RC) = \frac{TP}{TP + FN} \tag{3}$$

$$F1 - Measure\,(PR) = 2\,\frac{PR * RC}{PR + RC} \tag{4}$$

B. Testbed Setup

As illustrated in Fig. 7, the topology used in our tests consists of multiple PCs, including a Web server acting as the target, an attacker PC launching low-rate DDoS attacks, and several benign PCs generating legitimate traffic. The Mininet[2] emulator is used to model hosts and links. The P4 switches (2 spine and 6 leaf switches) and the SDN controller are each deployed in isolated containers with a 4-core CPU and 32 GB RAM on a Cisco UCS chassis. All hosts are interconnected via P4-enabled switches implementing the proposed BNN classifier, compiled using BMv2 [35].

To assess resilience against unknown low-rate DDoS attacks, the attacker PC launches the following attacks with carefully tuned parameters:

- **R.U.D.Y.**: Generates HTTP POST requests with very small payload fragments (40–100 bytes) sent at low rates (1–10 packets/s), with inter-packet intervals ranging from 200 ms to several seconds, sustaining connections for 150–300 s.
- **Slowloris**: Sends HTTP header fields in small chunks (40–80 bytes) at rates of 1–5 packets/s, with long idle times between transmissions to keep connections open for extended periods (150–300 s).
- **Sockstress**: Exploits TCP session persistence by opening connections with small advertised window sizes and low packet rates (5–50 packets/s), maintaining TCP states for 150–300 s.

To emulate multi-source attack scenarios, 10 to 300 concurrent malicious flows are generated using spoofed or distributed source addresses. In parallel, 5 to 20 benign hosts produce legitimate HTTP traffic at rates of 1–5 requests/s with packet sizes ranging from 500 to 1,500 bytes. Malicious flows used for evaluation represent approximately 30% of the test traffic and are excluded from the training phase.

C. BNN Classification Evaluation

1) Experimental Results and Interpretation: The BNN, DT, SVM, RF, and real-valued DNN models were trained on the GAN-generated dataset and evaluated on real-world data to assess the relevance of the generated samples. As shown in Table IV, the real-valued DNN achieved the highest accuracy ($AC = 0.9891$), followed by BNN ($AC = 0.9689$), SVM ($AC = 0.9682$), RF ($AC = 0.9260$), and DT ($AC = 0.9155$), with similar trends across the other metrics. All models achieved an

[2]http://mininet.org/

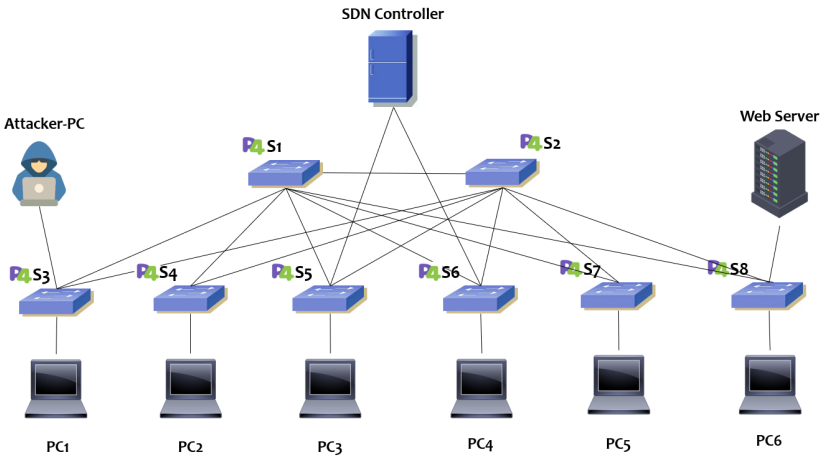


Fig. 7: Testbed Setup to evaluate GANShieldTrust.

TABLE IV: BNN Classification Evaluation.

Classification Model	Accuracy	Precision	Recall	F1
Real-value DNN	0.9891	0.9944	0.9843	0.9893
BNN	**0.9689**	**0.9382**	**0.9177**	**0.9278**
SVM	0.9682	0.9929	0.9502	0.9711
RF	0.9260	0.9314	0.9304	0.9308
DT	0.9155	0.9925	0.7711	0.8679

average classification time of 1–2 ms per flow with dynamic δ, confirming their suitability for near-real-time detection.

Although the DNN achieves the highest accuracy, the BNN provides the best overall trade-off between detection performance, computational efficiency, and data-plane deployability. The BNN achieves 96.89% accuracy while using 1024-bit binary weights, substantially reducing the memory and computational requirements associated with full-precision DNN parameters. This makes BNN inference more suitable for programmable data-plane environments, where hardware and processing resources are constrained. In contrast, full-precision DNNs and kernel-based SVMs generally require more computational resources and are less practical for direct deployment in programmable switches. As discussed in Section VI-D, although DNNs are effective for low-rate DDoS detection, their complexity makes direct deployment in programmable devices challenging. BNNs therefore provide a practical compromise between accuracy and efficiency, enabling lightweight, line-rate inference while maintaining strong detection performance, consistent with the integration of ML models into programmable hardware [36].

D. Controller-Based Attack Confirmation Evaluation

To evaluate controller-side attack confirmation, we measure its latency and effectiveness in validating reports from switches along the traffic path. When a switch detects a potential low-rate DDoS flow using its local BNN, it immediately blocks the flow and reports the event to the SDN controller. The controller aggregates reports from switches observing the same flow and confirms the attack when at least half report the same attack type, after which the mitigation decision is broadcast to all switches.

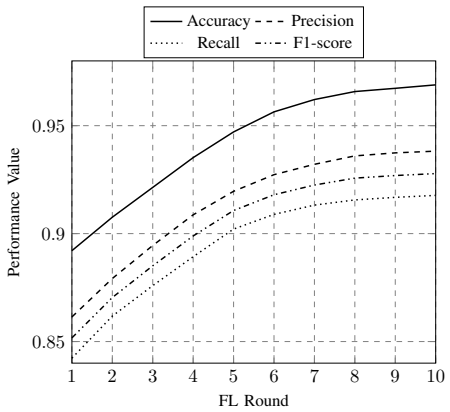


Fig. 8: Detection performance convergence across federated learning rounds.

The average switch reporting delay is 3.2 ms, while controller processing requires 2.5 ms. Confirmation latency averages 8.7 ms with 2–3 switches and 12.4 ms with 4 switches, keeping total mitigation latency below 20 ms. The limited increase with path length results from near-simultaneous reporting and parallel controller processing.

Local BNN classification achieves 93.2% accuracy, 90.2% precision, 88.3% recall, and an F1-score of 89.2%. Majority confirmation improves these metrics to 99.68% accuracy, 95.9% precision, 96.4% recall, and 96.1% F1-score, while reducing the false positive rate from 3.9% to 1.8%. These results demonstrate that cross-switch validation substantially improves detection reliability while maintaining near-real-time mitigation, particularly in typical 2–4-switch SDN paths.

E. Local BNN Training Evaluation

To evaluate the impact of federated BNN training on detection performance, the global model is evaluated after each federated learning round using a fixed validation dataset separate from training data. Accuracy, Precision, Recall, and F1-score are tracked to assess convergence and learning stability. The experiment comprises 10 federated rounds, each consisting of local switch training followed by controller-side aggregation. Results show progressive performance improvement, with the global model converging after approximately 8–10 rounds.

TABLE V: GAN-generated Dataset Evaluation with BNN.

Dataset	Accuracy	Precision	Recall	F1
GAN-generated	**0.9689**	**0.9382**	**0.9177**	**0.9278**
Original	0.8845	0.8946	0.8971	0.8958

TABLE VI: Federated Learning Evaluation.

Learning Method	Accuracy	Precision	Recall	F1
Federated Learning	**0.9689**	**0.9382**	**0.9177**	**0.9278**
Local Learning	0.9078	0.9108	0.9138	0.9122

Fig. 8 shows that most performance gains occur within the first five federated rounds as the model adapts to distributed traffic patterns. Subsequent rounds yield progressively smaller improvements because the aggregated model increasingly captures the common patterns shared across participating switches, leaving fewer significant errors to correct through further local updates. As the local models become more consistent, successive aggregation produces only marginal changes in the global parameters, resulting in stable performance by round 10. These results demonstrate effective knowledge aggregation across switches and stable convergence of the federated BNN.

F. GAN-generated Dataset Evaluation

This experiment evaluates the BNN classifier trained on the GAN-generated custom dataset against a BNN trained directly on the original dataset to evaluate whether the GAN-generated adversarial examples improve detection.

1) Experimental Results and Interpretation: As shown in Table V, the superior performance of the GAN-generated model compared to the original model is attributed to the GAN's ability to generate flow features that highly mimic those extracted from real attack traffic, empowering better resiliency of the BNN model and helping to avoid overfitting.

G. Federated Learning Evaluation

In this scenario, we demonstrate the benefits of using federated learning in detecting low-rate DDoS attacks. We compare the detection performance between two BNN detection schemes: the first is the design employed in this work without modification. In the second, we restrict the learning data used in GAN to locally-extracted flow features; i.e., from the local switch only, and then we train the BNN model with the synthetic dataset generated using GAN model. The attack detection process remains the same, meaning that an attack is confirmed only after the controller's approbation.

1) Experimental Results and Interpretation: Table VI shows a clear performance improvement with federated learning, with accuracy increasing from 0.9078 for local training to 0.9689 after federated aggregation, alongside consistent improvements across the other metrics. This gain results from sharing learning across switches while preserving data locality, allowing the aggregated BNN to benefit from diverse traffic observations and progressively reduce local classification errors.

To further address non-IID traffic distributions across switches, we evaluated a DBSCAN-based clustering mechanism that groups switches with similar traffic profiles before federated aggregation. DBSCAN was selected because it can identify clusters based on similarity without requiring a predefined number of clusters, while also handling irregular cluster structures and switches with distinct traffic profiles as outliers. Each switch periodically reports a compact traffic profile, defined as a small set of statistical features summarizing its observed traffic characteristics ($\approx$ 64–96 bytes), and clustering is performed at the controller with negligible overhead ($< 0.05\%$ control traffic and $< 15ms$ processing for up to 32 switches). Clustering improves model convergence stability and detection consistency by limiting conflicting updates from heterogeneous traffic environments. Compared with standard federated aggregation, clustered federated learning reduced convergence rounds by $\approx 35\%$ and improved robustness to localized traffic variations. Experimental results show measurable gains in detection performance. Without clustering, the global model achieved lower detection accuracy (Table VII), whereas DBSCAN-based clustering improved performance by better aligning model updates from switches with similar traffic characteristics. These gains indicate that clustering can reduce model drift caused by non-IID data and improve the consistency of the learned model across heterogeneous network environments.

TABLE VII: Federated Learning Clustering Evaluation.

Clustering	Accuracy	Precision	Recall	F1
DBSCAN-based	**0.9689**	**0.9382**	**0.9177**	**0.9278**
Raw	0.8500	0.8100	0.7950	0.8020

H. Secure Federated Learning Evaluation

To assess robustness against weight poisoning, we assume that an attacker compromises *Switch1* and replaces the averaged weights with malicious ones during secure federated learning. The attacker is assumed to control fewer than half of the switches, preserving majority agreement for attack detection and federated learning. Experimental results show that all poisoning attempts are detected by comparing the weight hash reported by each node with the hash sum computed by the miner; any mismatch indicates an attack.

I. Overall Performance Evaluation

After evaluating each component, we now assess the overall performance of our solution compared to other SDN-based, deep learning approaches that detect unknown low-rate DDoS attacks. The selected solutions are: [9], [37], [38], and also [7], [24] for their P4-based feature extraction, though they target general DDoS. Table VIII summarizes their performance and features. GANShieldTrust stands out by enabling distributed training, fast P4-based detection, GAN-based robust training, secure federated learning, and global learning, offering greater practicality and resilience than partial approaches. However, a direct numerical comparison should be interpreted with caution. The reviewed solutions differ significantly in the complexity of the attacks considered, the flow features used, the model architectures, and the evaluation setups. In addition, several metrics reported in our work are not evaluated or disclosed in some of the compared studies. Therefore, Table VIII should be

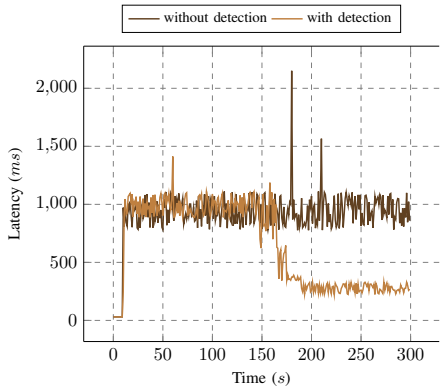


Fig. 9: R.U.D.Y. Attack Mitigation Latency.

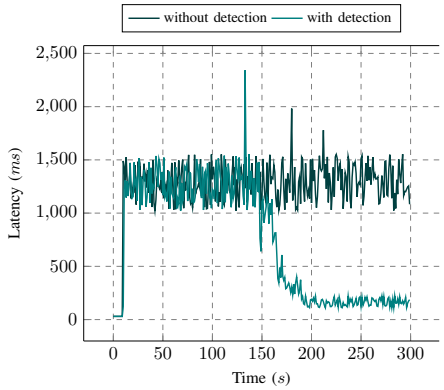


Fig. 10: Slowloris Attack Mitigation Latency.

viewed primarily as an architectural and capability comparison rather than a strictly equivalent quantitative benchmark.

J. Attack Mitigation Latency Evaluation

In this scenario, we evaluate the web server's response time when the server is under attack, compared to the response time when our solution is activated on the switches. The attacks considered are R.U.D.Y., Slowloris, and Sockstress.

Fig.9, Fig.10, and Fig. 11 show the latency over time caused by R.U.D.Y., Slowloris, and Sockstress attacks, respectively. Darker colors represent latency without GANShieldTrust; lighter colors represent latency with it active.

All attacks start at $10s$, causing latency spikes: $\sim 1,000ms$ for R.U.D.Y., $1,000$–$1,500ms$ for Slowloris and Sockstress. Sockstress, being TCP-based (Layer 4), affects all services, not just HTTP.

Mitigation begins at $150s$, blocking malicious traffic within 30–$40s$. We tuned the δ window to increase with latency, ensuring timely detection. Once blocked, both latency and δ return to normal values.

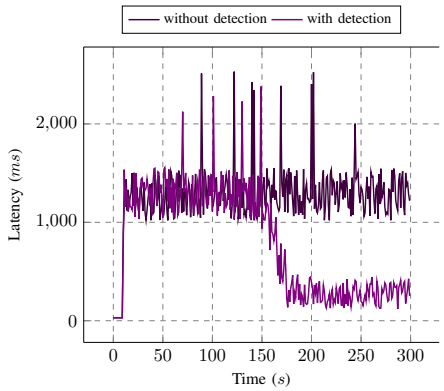


Fig. 11: Sockstress Attack Mitigation Latency.

We also compare the latency overhead introduced by the solution and the model convergence time with existing works. Among the considered approaches, only [7] reports explicit latency measurements, showing an average processing latency of approximately 0.4ms, while the model convergence time is not provided. For other solutions, including [9], [37], [38], and [24], neither the latency overhead nor the model convergence time is explicitly reported, preventing a direct quantitative comparison.

In contrast, GANShieldTrust explicitly evaluates both aspects. It introduces low and bounded latency through P4-based BNN classification at the data plane, and achieves stable convergence through iterative federated learning enhanced with GAN-generated data. This highlights its suitability for real-time SDN deployments where both inference latency and convergence behavior are critical.

K. δ Value Impact on Detection

This section evaluates how different δ values impact detection accuracy and delay. δ defines the sampling interval for flow feature extraction. Smaller δ yields faster detection but lower accuracy due to insufficient flow representation. Larger δ improves accuracy but slows response. To balance both, δ is adapted dynamically: it increases when response delay rises (signaling a potential attack) and decreases when the delay normalizes.

Figs. 12 and 13 show that accuracy is lower between $0.1s$–$2.9s$, then improves beyond $3s$, peaking at 96.89% as δ aligns with attack-induced delays.

Compared to existing solutions, GANShieldTrust achieves faster and more adaptive detection for low-rate DDoS attacks. While [7] reports a 2ms detection time for 95% of traffic, it targets standard intrusion attacks (e.g., botnets) rather than low-rate DDoS. In contrast, [9] requires 10s and 45s to detect Hulk and Slowloris attacks, respectively. Other SDN-based solutions such as [37] and [38] do not report detection time. Although [24] achieves a 0.4ms classification time using SVM, it is limited to TCP flood attacks and relies on simplified traffic features. Overall, GANShieldTrust offers a more comprehensive and realistic trade-off between detection speed, attack coverage, and feature richness for low-rate DDoS mitigation.

L. Attack From Multiple Sources

Each switch in GANShieldTrust independently detects and mitigates low-rate DDoS attacks, including multi-source attacks, through per-flow feature extraction and BNN classification in the P4 pipeline. Multiple simultaneous attacks can be generated through IP spoofing or different flows and emulated using tools such as Scapy or Hping3 in controlled environments such as Mininet. Fig. 14 shows that the first switch maintains approximately 96% accuracy up to $\sim$ 275 flows before decreasing to approximately 85%, while the second and third switches progressively compensate as earlier switches saturate. This distributed operation provides resilience to increasing attack volumes and individual switch overloads.

Detection latency remains bounded because flows are classified independently in the P4 pipeline, with excess malicious traffic progressively handled by upstream switches. Model training and federated aggregation are performed offline at the controller and remain outside the critical

TABLE VIII: Comparative table between GANShieldTrust and similar solutions with their proposed features.

Reference	Classifier	Distributed Training	Line-speed Detection	Adversarial Training	Secure Weight Exchange	Data Privacy	Detection Accuracy
GANShieldTrust	**BNN**	**P4**	**P4**	**GAN**	**Blockchain**	**Federated Learning**	**0.9689**
Benzaïd et al. [9]	MLP	No	No	FGSM	No	No	0.9965
Sun et al. [37]	Hybrid CNN-GRU	No	No	No	No	No	0.9950
Nugraha et al. [38]	CNN-LSTM	No	No	No	No	No	0.9900
Qin et al. [7]	BNN	P4	P4	No	No	No	0.9450
Musumeci et al. [24]	RF, KNN, and SVM	No	No	No	No	No	0.9850 (RF)

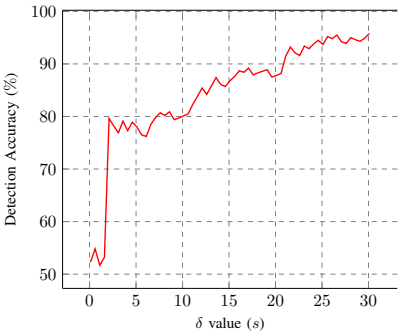


Fig. 12: Detection Accuracy over Variable δ Value

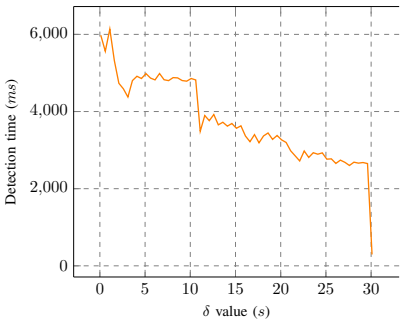


Fig. 13: Detection Time over Variable δ Value

data path; although aggregation and verification time increases with the number of switches, periodic training does not affect real-time classification or mitigation. Thus, GANShieldTrust maintains scalable and consistent inline performance under large-scale, multi-source attacks.

M. Overhead Evaluation

1) Computational Overhead: Fig. 15 shows the computational overhead at the switch level. During traffic classification, the average CPU load is around 39%, thanks to BNNs' lightweight design enabling line-speed

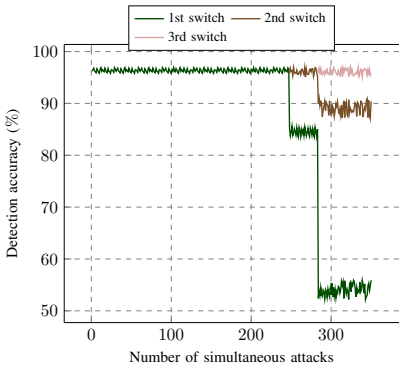


Fig. 14: Number of attacks' impact on accuracy on various switches

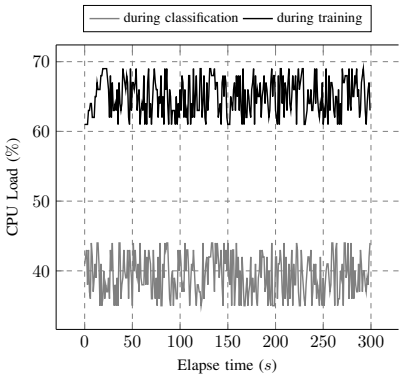


Fig. 15: Computational Overhead introduced by GAN-ShieldTrust

processing. During training, CPU usage increases to 69% due to the use of real-value weights for improved accuracy.

These results are partly due to the use of BMv2, the reference P4 software switch, which is well suited for development and testing but has significant performance limitations compared with production-grade, ASIC-based programmable switches such as Intel Tofino and Tofino 2 [39]. Running on general-purpose CPUs, BMv2 provides lower throughput and higher latency due to limited parallelism, memory constraints, and software-based processing overhead [40]. Its flexibility and programmable P4 pipeline therefore introduce higher computational overhead, resulting in measured loads that may exceed those expected on optimized hardware switches used in high-performance networks.

2) Secure Federated Learning Overhead: Each node broadcasts a 256-bit hash of its binary weight vector, while the elected miner distributes the block hash for integrity verification. Differential Privacy adds negligible overhead, with Secure Federated Learning traffic accounting for only $\sim 1.2\%$ of total traffic.

ACKNOWLEDGMENT

The research work is supported in part by the Federal Ministry of Research, Technology, and Space (BMFTR), Germany, through the Project 6GEM+ under Grant 16KIS2411; and in part by the European Union's Horizon Europe research and innovation programme under the 6G-Path project (Grant No. 101139172).

VIII. CONCLUSION

In this paper, we introduced GANShieldTrust, a fully distributed DL-based solution that counters adversarial and unknown low-rate DDoS attacks. It combines GAN for data synthesis, BNN for fast detection, federated learning for distributed learning, and blockchain for integrity and

privacy. We evaluated it on an SDN testbed using a train-on-synthetic-and-test-on-real approach with R.U.D.Y., Slowloris, and Sockstress attacks. We assessed each component individually, then compared our full solution to existing SDN-based approaches. GANShieldTrust demonstrated strong detection accuracy and outperformed others in speed, data integrity, resilience, privacy, and scalability.

Future work will explore advanced GAN models (e.g., cGAN, WGAN), distributed federated learning among switches (removing central aggregation), and real-world deployment on programmable ASICs like Intel Tofino to evaluate performance in production-grade environments.

REFERENCES

[1] A. Kuzmanovic and E. W. Knightly, "Low-rate tcp-targeted denial of service attacks: The shrew vs. the mice and elephants," in *Proc. of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM'03)*, 2003, p. 75–86.

[2] W. Zhijun, L. Wenjing, L. Liang, and Y. Meng, "Low-rate dos attacks, detection, defense, and challenges: A survey," *IEEE Access*, vol. 8, pp. 43 920–43 943, 2020.

[3] V. D. M. Rios, P. R. M. Inácio, D. Magoni, and M. M. Freire, "Detection and mitigation of low-rate denial-of-service attacks: A survey," *IEEE Access*, vol. 10, pp. 76 648–76 668, 2022.

[4] M. Macas, C. Wu, and W. Fuertes, "A survey on deep learning for cybersecurity: Progress, challenges, and opportunities," *Computer Networks*, vol. 212, p. 109032, Jul. 2022.

[5] U. Sabeel, S. S. Heydari, H. Mohanka, Y. Bendhaou, K. Elgazzar, and K. El-Khatib, "Evaluation of deep learning in detecting unknown network attacks," in *2019 International Conference on Smart Applications, Communications and Networking (SmartNets)*, 2019, pp. 1–6.

[6] M. N and J. B., "A deep learning based http slow dos classification approach using flow data," *ICT Express*, vol. 7, 09 2020.

[7] Q. Qin, K. Poularakis, K. K. Leung, and L. Tassiulas, "Line-speed and scalable intrusion detection at the network edge via federated learning," in *2020 IFIP Networking Conference (Networking)*, 2020, pp. 352–360.

[8] M. P. Novaes, L. F. Carvalho, J. Lloret, and M. L. Proença, "Adversarial deep learning approach detection and defense against ddos attacks in sdn environments," *Future Generation Computer Systems*, vol. 125, pp. 156–167, 2021.

[9] C. Benzaïd, M. Boukhalfa, and T. Taleb, "Robust self-protection against application-layer (d)dos attacks in sdn environment," in *2020 IEEE Wireless Communications and Networking Conference (WCNC)*, 2020, pp. 1–6.

[10] M. N. Ali, M. Imran, M. S. u. din, and B.-S. Kim, "Low rate ddos detection using weighted federated learning in sdn control plane in iot network," *Applied Sciences*, vol. 13, no. 3, 2023.

[11] A. A. Alashhab, M. S. M. Zahid, M. Abdullahi, and M. S. Rahman, "Real-time detection of low-rate ddos attacks in sdn-based networks using online machine learning model," in *Proc. of 7th Cyber Security in Networking Conf. (CSNet)*, Oct. 2023, pp. 95–101.

[12] D. Tang, S. Wang, B. Liu, W. Jin, and J. Zhang, "Gasf-ipp: Detection and mitigation of ldos attack in sdn," *IEEE Transactions on Services Computing*, vol. 16, no. 5, pp. 3373–3384, 2023.

[13] P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese, and D. Walker, "P4: Programming Protocol-independent Packet Processors," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 88 – 95, Jul. 2014.

[14] K. B. Adedeji, A. M. Abu-Mahfouz, and A. M. Kurien, "Ddos attack and detection methods in internet-enabled networks: Concept, research perspectives, and challenges," *Journal of Sensor and Actuator Networks*, vol. 12, no. 4, 2023. [Online]. Available: https://www.mdpi.com/2224-2708/12/4/51

[15] M. Mittal, K. Saluja, and S. Behal, "Deep learning approaches for detecting ddos attacks: A systematic review," *Soft Computing*, vol. 27, p. 13039–13075, Sept. 2023.

[16] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Commun. ACM*, vol. 63, no. 11, p. 139–144, Oct. 2020.

[17] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 12, pp. 4217–4228, Dec. 2021.

[18] R. Chauhan and S. Shah Heydari, "Polymorphic adversarial ddos attack on ids using gan," in *2020 Int. Symp. on Networks, Computers and Communications (ISNCC)*, 2020, pp. 1–6.

[19] O. A. Wahab, A. Mourad, H. Otrok, and T. Taleb, "Federated Machine Learning: Survey, Multi-Level Classification, Desirable Criteria and Future Directions in Communication and Networking Systems," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 2, pp. 1342 – 1397, Secondquarter 2021.

[20] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[21] C. Benzaïd, T. Taleb, A. Sami, and O. Hireche, "FortisEDoS: A Deep Transfer Learning-Empowered Economical Denial of Sustainability Detection Framework for Cloud-Native Network Slicing," *IEEE Trans. on Dependable and Secure Comput.*, vol. 21, no. 4, pp. 2818–2835, July-Aug. 2024.

[22] X. Liu, J. Ren, H. He, Q. Wang, and C. Song, "Low-rate ddos attacks detection method using data compression and behavior divergence measurement," *Computers & Security*, vol. 100, p. 102107, 2021.

[23] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks," in *Advances in Neural Information Processing Systems*, vol. 29, 2016.

[24] F. Musumeci, V. Ionata, F. Paolucci, F. Cugini, and M. Tornatore, "Machine-learning-assisted ddos attack detection with p4 language," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020, pp. 1–6.

[25] M. L. Adjou, C. Benzaïd, and T. Taleb, "Topotrust: A blockchain-based trustless and secure topology discovery in sdns," in *2022 International Wireless Communications and Mobile Computing (IWCMC)*, 2022, pp. 1107–1112.

[26] M. Yassine, S. Mounir, K. El Gholami, Q. Youssef, and Y. Sadqi, "A comprehensive survey on sdn security: threats, mitigations, and future directions," *Journal of Reliable Intelligent Environments*, vol. 9, p. 201–239, Jun. 2023.

[27] R. Scott, B. Oliver, M. Stu, and C. Sean, "Zero trust architecture," National Institute of Standards and Technology, Tech. Rep. NIST Special Publication 800-207, Aug. 2020.

[28] "Cic dos dataset (2017)," https://www.unb.ca/cic/datasets/dos-dataset.html, accessed: 2023-01-14.

[29] "Low rate ddos attack dataset," https://www.kaggle.com/datasets/keval17/ddosattackdetection, accessed: 2023-01-14.

[30] "Cse-cic-ids2018 on aws," https://www.unb.ca/cic/datasets/ids-2018.html, accessed: 2023-01-14.

[31] N. Shibahara, M. Koibuchi, and H. Matsutani, "Performance Improvement of Federated Learning Server using Smart NIC," in *Proc. of 11th International Symposium on Computing and Networking Workshops (CANDARW)*, Nov. 2023, pp. 165–171.

[32] "Hpe aruba networking cx 10000 switch series quickspecs," https://www.hpe.com/us/en/collaterals/collateral.a50004267enw.html, Accessed: 2026-03-31.

[33] "Amd dpu-accelerated networking for ai and services in high-performance data centers," https://www.ciscolive.com/c/dam/r/ciscolive/emea/docs/2025/pdf/PARDCN-1333.pdf, Accessed: 2026-03-31.

[34] K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, T. Q. S. Quek, and H. Vincent Poor, "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3454–3469, 2020.

[35] "BEHAVIORAL MODEL (bmv2)," https://github.com/p4lang/behavioral-model.

[36] Z. Xiong and N. Zilberman, "Do switches dream of machine learning? toward in-network classification," in *Proceedings of the 18th ACM Workshop on Hot Topics in Networks (HotNets'19)*, 2019, p. 25–33.

[37] W. Sun, S. Guan, P. Wang, and Q. Wu, "A hybrid deep learning model based low-rate dos attack detection method for software defined network," *Trans. Emerg. Telecommun. Technol.*, vol. 33, no. 5, may 2022.

[38] B. Nugraha and R. N. Murthy, "Deep learning-based slow ddos attack detection in sdn-based networks," in *2020 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, 2020, pp. 51–56.

[39] Intel and Kaloom, "Intel, kaloom create p4-programmable network solutions," Intel Network Builders, White Paper, 2020, accessed: 2024-04-27. [Online]. Available: https://networkbuilders.intel.com/docs/networkbuilders/intel-kaloom-create-p4-programmable-network-solutions.pdf

[40] I. Tsareva, D. Scholz, and S. Gallenmüller, "Taxonomy of the performance of p4 targets," in *Proc. of the Seminar Innovative Internet Technologies and Mobile Communications (IITM)*, 2021, pp. 25–29.



Mohamed Lamine Adjou received the M.S. degree in Networks and Distributed Sytems from the University of Science and Technology Houari Boumediene (USTHB), Algiers, Algeria. He is currently pursuing the Ph.D. degree in network and security at USTHB. His research interests include software-defined networking (SDN), network security, artificial intelligence and machine learning for network intrusion detection, and secure and resilient network architectures. He is also experienced in teaching advanced networking technologies, particularly routing protocols, including service provider routing protocols and enterprise network infrastructures.



Chafika Benzaïd is currently a senior research fellow at University of Oulu, Finland. Between Nov. 2018 and Dec. 2021, she was senior researcher at Aalto University. Before that, she worked as an associate professor at University of Sciences and Technology Houari Boumediene (USTHB). She holds Engineer, Magister and "Doctorat en Sciences" degrees from USTHB. Her research interests lie in the field of 5G/6G, SDN, Network Security, AI Security, and AI/ML for zero-touch security management. She is an ACM professional & IEEE ComSoc member.



Tarik Taleb received the B.E. degree with distinction in Information Engineering and the M.Sc. and Ph.D. degrees in Information Sciences from Tohoku University, Sendai, Japan, in 2001, 2003, and 2005, respectively. He is currently a Full Professor at Ruhr University Bochum (RUB), Germany, where he leads research activities on next-generation mobile and distributed systems. Prior to joining RUB, he was a Professor at the University of Oulu, Finland (2018–2023), and an Associate Professor at Aalto University, Finland (2014–2021). Earlier in his career, he served as a Senior Researcher and 3GPP Standards Expert at NEC Europe Ltd., Heidelberg, Germany, where he contributed to the evolution of mobile network architectures and standardization activities. Before joining NEC, he was an Assistant Professor at the Graduate School of Information Sciences, Tohoku University, Japan, working in a research laboratory fully funded by KDDI. He also held a Research Fellowship at the Intelligent Cosmos Research Institute, Japan, from 2005 to 2006. Prof. Taleb is widely recognized for his pioneering contributions to mobile network softwarization, network slicing, cloud-edge continuum management, and autonomous networking. His current research interests include autonomous network and service management, edge-cloud continuum systems, network softwarization and slicing, software-defined security, and AI-native communication networks.