f-AnoGAN for Unsupervised Attack Detection in SDN Environment

Vitor Gabriel da Silva Ruffo, Luiz Fernando Carvalho, Jaime Lloret, *Senior Member, IEEE*, and Mario Lemes Proença Jr

***Abstract*—Network management solutions remain essential for proper network service delivery. The software-defined networking (SDN) paradigm brought flexibility and programmability to today's large-scale networks, easing their governance. Another critical factor in the quality of network services is network security for protection against cyberattacks. This work proposes an unsupervised volume anomaly detection and mitigation system for securing SDN environments. We implement a fast AnoGAN (f-AnoGAN) to model legitimate user behavior and identify outlier samples. The generative network is trained on a low-dimensional representation of network traffic to reduce computational overhead. The f-AnoGAN model performance is further investigated through hyperparameter tuning and ablation study. The security system is evaluated on four public datasets: Orion, CIC-DDoS2019, CIC-IDS2017, and TON_IoT. We implement state-of-the-art alternative models for comparison analysis, namely Autoencoder, BiGAN, and FID-GAN. The f-AnoGAN presents improved class separation capacity and anomaly identification performance compared to the other models. The anomaly mitigation module can drop between 95% and 99% of malign traffic, supporting network resilience and correct functioning.**

***Index Terms*—Unsupervised learning, generative adversarial network, f-anogan, anomaly detection, software-defined network.**

I. Introduction

THE society strongly relies on the efficient delivery of online services. Examples include communication applications, digital banking and shopping, and AI assistance. This dependence has motivated the continuous development of networking technology to promote quality of service and resilience. The TCP/IP networks have grown exponentially in size, traffic, and transfer speed. The networks' evolution motivates the continuous advancement of automatic and proactive management practices [1], [2], [3], [4].

The software-defined networking (SDN) paradigm has brought many advantages to network management, such as flexibility and programmability. It separates the data and control planes, simplifying establishing control rules, fulfilling dynamic requisites, and policy enforcement through a unique protocol [5], [6], [7], [8]. Security is another management concern that network service providers must consider. Confidentiality, integrity, and availability are fundamental to running reliable networked solutions. Malicious agents may attack them by accessing, altering, or blocking confidential data [9].

The scientific community has been studying network attack detection algorithms to identify cyber threats automatically. A popular solution is regarded as a Network Intrusion Detection System (NIDS) [10], [11], [12]. This system is installed in the network to capture and analyze all traffic traversing specific nodes. NIDS are classified into two categories by the employed detection paradigm. These are signature-based and anomaly-based. The former compares incoming traffic with previously labeled attack samples. Any match is considered abnormal and reported to network administration for further inspection. The anomaly-based implementation models legitimate users' behavior, interpreting any considerable deviations in the observed traffic as anomaly occurrences.

Deep Learning (DL) models have shown remarkable results in the realm of anomaly detection compared to alternative approaches, providing state-of-the-art performance [13], [14], [15]. Many DL models can be applied to distinct data types or solve different pattern recognition problems. For example, vanilla deep neural networks are standard models for extracting generic representative features from data and deriving inferences from them. Recurrent neural networks are employed to learn the temporal correlation among time-distant samples. Convolutional neural networks often apply to capture implicit spatial patterns in their input. Generative Adversarial Networks (GAN) are trained to approximate the learning distribution p_g to the training data distribution p_r [16].

Neural networks can be trained for anomaly detection using supervised or unsupervised learning paradigms. The first setup demands labeled legitimate and attack data. The model commonly learns to map the incoming traffic to its correct classification. In the unsupervised setup, the model strives to uncover hidden structures and relations from unlabeled data, using the

Received 17 October 2024; revised 1 April 2025; accepted 4 April 2025. Date of publication 10 April 2025; date of current version 27 June 2025. This work was supported in part by CAPES, Brazil, due to the concession of scholarships, in part by the National Council for Scientific and Technological Development (CNPq) of Brazil under Project 306397/2022-6, in part by the Superintendence of Science, Technology, and Higher Education (SETI), in part by Araucaria Foundation, and in part by the State University of Londrina (PROPPG). Recommended for acceptance by Dr. Bo Yang. *(Corresponding author: Jaime Lloret.)*

Vitor Gabriel da Silva Ruffo and Mario Lemes Proença Jr are with the Computer Science Department, State University of Londrina, Londrina, Paraná 86057-970, Brazil.

Luiz Fernando Carvalho is with the Federal Technology University of Paraná, Apucarana, Paraná 86812-460, Brazil.

Jaime Lloret is with the Integrated Management of Coastal Areas Research Institute, Universitat Politecnica de Valencia, 46730 Gandia, Spain (e-mail: jlloret@dcom.upv.es).

Digital Object Identifier 10.1109/TNSE.2025.3558936

acquired knowledge for anomaly detection. Recent research has shown that the success of many deep learning-based NIDS experimental studies is still tied to the supervised paradigm [17], [18], [19], [20]. These types of systems naturally have a high implementation cost in real-world scenarios. They require historical data to be captured and thoroughly labeled by human experts for their configuration [21], [22]. Therefore, the underexplored, unsupervised-based NIDS is a promising research direction that reduces NIDS deployment overhead in non-laboratory environments.

This research paper proposes an unsupervised volume-based anomaly detection and mitigation system for software-defined network environments, promoting security against cyberattacks without human intervention. Given the provided management flexibility, the SDN paradigm is more suitable to support large-scale, ever-increasing current networks than the traditional one. We, therefore, design our system to operate in such dynamic contexts that reflect the future of real-world networking environments. The solution models legitimate traffic behavior by training a fast AnoGAN (f-AnoGAN) network on entropy features. The generative network adversarially learns the data distribution, enabling fast encoding and decoding of input samples. The detection module applies sample reconstruction error and discriminative scoring to identify outlier communication patterns. The mitigation module runs in parallel and is invoked to block newly found anomalous traffic, automatically protecting the network resources. The f-AnoGAN is utilized in the system's core rather than other generative models since it learns using the Wasserstein distance, bringing training stability and mode collapse reduction. This generative network also allows for the implementation of an unsupervised anomaly detection algorithm leveraging both the generator and discriminator.

The contributions are the following:

- Propose an autonomous algorithm for unsupervised volume-based anomaly detection and mitigation;
- Introduce an entropy-based f-AnoGAN model for fast data reconstruction;
- Investigate the generative model behavior through hyperparameter tuning and ablation study;
- Conduct experimental validation on public benchmark datasets, namely Orion [23], CIC-DDoS2019 [24], CIC-IDS2017 [25], and TON_IoT [26];
- Run a comparison analysis between the proposed system and state-of-the-art similar models, such as Autoencoder, BiGAN [27], and FID-GAN [21].

The remaining sections are organized in the following way. Section II introduces background concepts. Section III discusses the related works, highlighting our proposal's contributions. Section IV describes the proposed system's implementation details. Section V presents the experimental setup and results that support the validity of our solution. Section VI elaborates on the conclusions and future work.

II. Background

The Generative Adversarial Network is a deep neural network introduced by Goodfellow et al. [28]. The GAN model is trained under the unsupervised learning paradigm to solve implicit density estimation. A well-trained generative network can generate new, synthetic data samples that fit its training data's probability distribution P_r [29]. The model's distribution learning capacity has been leveraged for network behavior modeling.

The scientific community commonly trains GANs to generate rare classes of network traffic samples [30]. Data generation is usually aimed at improving the training quality of traffic classification algorithms. The network anomaly detection field has also applied GAN to define a network behavior baseline [16]. Outlier traffic samples can be identified when their observed behavior diverges from the calculated baseline. The following subsections present a theoretical discussion on the Generative Adversarial Network as well as some of its established variations, including Wasserstein GAN with Gradient Penalty and Bidirectional GAN.

A. Generative Adversarial Network

The internal architecture of a Generative Adversarial Network comprises two neural networks, referenced as generator G and discriminator D. These networks are set to compete against each other in a min-max optimization game. The generator represents a function for mapping low-dimensional noise vectors z to high-dimensional data samples x. The noise vectors fed to G are sampled from a distribution P_z over the latent space. The discriminator is a binary classifier for separating synthetic data generated by G from real data composing the training data set. The distribution of generated data and real data are indicated as P_g and P_r, respectively [31], [32].

The Generator and the Discriminator are trained concurrently using opposite objective functions. A reduction in G's cost function causes an increase in the D's cost in the same proportion, and vice-versa. Their cost function V is defined in (1), being composed of the sum of two subfunctions A (2) and B (3). The symbol $\mathbb{E}$ denotes the expected value or average of its operand. The terms $D(x)$ and $D(G(z))$ indicate the Discriminator's output for a real sample x and a synthetic one $G(z)$, respectively. The function A(D) then provides the average logarithm of the Discriminator's output for real data. The function B(D, G) specifies the average complement of the Discriminator's output for fake data on a logarithmic scale.

During training, D's parameters are tweaked to maximize V. A and B are both increased by approximating $D(x)$ to 1 and $D(G(z))$ to 0, aiding the Discriminator to classify its input correctly. Conversely, G's parameters are adjusted to minimize V. A is ignored for tuning G since the latter's parameters do not influence the value of the function. The function B is minimized by inducing D to classify the synthetic data $G(z)$ as 1, helping the Generator to produce convincing samples. At the end of the learning process, the Generator network may approximate the generated data distribution P_g to the training data distribution P_r. In this scenario, G can completely fool D by generating synthetic data indistinguishable from the real ones [28], [33].

$$\min_G \max_D V(D,G) = A(D) + B(D,G) \tag{1}$$

$$A(D) = \mathbb{E}_{x\sim P_r(x)}[\log D(x)] \tag{2}$$

$$B(D,G) = \mathbb{E}_{z\sim P_z(z)}[\log(1 - D(G(z)))] \tag{3}$$

TABLE I
RELATED WORKS COMPARISON

Work	Model	Environment	Benchmark	Configuration paradigm	Thresholding specification	Mitigation
[38]	M/M/c queue	SDN	Emulated data	N/A	N/A	N/A
[39]	GAN, DCGAN, WGAN-GP	SDN	Emulated data, CIC-DDoS2019	supervised	✓	✓
[40]	AE	Traditional	NSL-KDD	unsupervised	✓	✗
[41]	BiGAN	Vehicular SDN	KDD99, NSL-KDD	unsupervised	✗	✗
[21]	FID-GAN	CPS	KDD99	unsupervised	✗	✗
[22]	DVGAN	CPS	SWAT, WADI, NSL-KDD	unsupervised	✗	✗
[42]	GAN, LSTM	Fog	NSL-KDD	unsupervised	✗	✗
[43]	Attention GAN	IoT	SWMRU, KDD99, HomeC	supervised	✗	✗
[44]	DCGAN	IoT	CIC-DDOS2019, CSE-CIC-IDS2018	supervised	N/A	✗
[45]	AE, BiGAN	IoT	IoT-23	supervised	N/A	✗
[46]	DIWGAN	WSN	NSL-KDD, CIC-IDS2017	supervised	✗	✗
This work	f-AnoGAN	SDN	Orion, CIC-DDoS2019, CIC-IDS2017, TON_IoT	unsupervised	✓	✓

Inherent problems in the initial conception of GAN have been reported in the literature. For instance, the Generator network may fail to approximate P_g to P_r due to the training procedure's intrinsic instability. Another common issue is mode collapse, referring to G learning only subparts of P_r. Consequently, there is a lack of diversity in the generated samples [16], [34].

B. Wasserstein GAN With Gradient Penalty

The Wasserstein GAN with Gradient Penalty (WGAN-GP) is a well-known example of GAN variation aimed at tackling training instability and reducing the degree of mode collapse [35]. The model applies the Wasserstein distance as its objective function, promoting more stable training as demonstrated through theoretical and experimental analysis. The WGAN-GP conception requires that the Discriminator network be a 1-Lipschitz function. A gradient penalty regularization term is included in the objective function to enforce the Lipschitz constraint on the Discriminator D.

C. Bidirectional GAN

The Bidirectional GAN (BiGAN) is a generative model that introduces an Encoder E within the GAN structure [27]. The BiGAN adapts the original objective function to consider the encoding network, which, as well as the Generator, is focused on fooling the Discriminator. The E network's parameters are adjusted to learn the inverse function of Generator G, i.e., mapping from data space to noise space. At the end of the training, the Encoder and Generator can collaboratively code and decode data samples such that $G(E(x)) \approx x$.

III. RELATED WORKS

The scientific community has been working towards developing progressively better intrusion detection systems [36], [37]. Table I summarizes the main features of the works reviewed in this section. Yuan et al. [38] proposed a DDoS defense mechanism based on CPU allocation management for the SDN environment's data and control planes. The authors state that the network quality of service can resist a DDoS attack by allocating sufficient processing resources for the switches and controller. They apply a queueing theory-based mathematical model to estimate the minimum required CPUs assigned to the networking devices. The experiments are run in a virtual machine using Mininet to emulate the SDN network and Ryu to implement the SDN controller.

Zacaron et al. [39] presented an evaluation of distinct versions of the Generative Adversarial Network applied to intrusion detection and mitigation in software-defined networks. The authors compare the vanilla GAN, the Deep Convolutional GAN, and the Wasserstein GAN with Gradient Penalty for anomaly detection by processing volumetric IP flow features. The discriminator network is leveraged in all models to calculate anomaly scores and separate benign samples from malign ones. The experimental scenarios are represented by traffic data collected from an emulated SDN and the widely-known CIC-DDoS2019 dataset, presenting promising results. Nevertheless, the system depends on a labeled validation dataset for tuning important hyperparameters, such as the classification threshold, which may burden its installation in real environments.

Choi et al. [40] proposed an unsupervised NIDS based on the Autoencoder network for calculating anomaly scores. Multiple variations of the traditional AE are trained and evaluated using an unlabeled collection of normal and abnormal data. A decision

threshold is defined based on the mean and standard deviation of training data, assuming they follow a normal distribution. The authors, however, do not prove the generality of their proposal since their experimental setup comprises a single outdated dataset, namely, NSL-KDD.

Shu et al. [41] described a decentralized intrusion detection system designed for multi-controller, SDN-based vehicular networks. A distributed BiGAN model is collaboratively trained on a cloud server and many subnetwork controllers aiming to achieve efficiency and effectiveness. The system detects anomalies by considering sample reconstruction errors and discriminating scores. Nonetheless, the authors validated their work on non-representative datasets, i.e., KDD99 and NSL-KDD. There is also a lack of guidance regarding the definition of the inference threshold.

Araujo-Filho et al. [21] introduced an unsupervised NIDS for securing cyber-physical systems, namely FID-GAN. The system uses the fog paradigm to leverage its computing resource availability and reduce detection latency. The system models legitimate traffic behavior by training a vanilla GAN implemented with LSTM units and an encoder to learn the inverse transformation of the generator network. The proposed detection algorithm utilizes the reconstruction error and discriminator output to calculate an incoming sample's anomalous degree. The system still lacks the specification of an automatic, label-independent threshold, which is a fundamental step for implementing such unsupervised systems in a real environment. The considered network dataset KDD99 may include dated traffic and attack patterns, representing an unsuitable validation scenario.

Sun et al. [22] introduced an unsupervised dual variational GAN for detecting anomalies in industrial cyber-physical systems. The system processes sensor time-series data using long short-term memory units in the generative model internal networks. A linear combination of reconstruction and discrimination scores gives the samples' anomaly degrees. The experimentation is performed on SWAT, WADI, and NSL-KDD datasets, comparing against models such as MAD-GAN, EGAN, and GAN-AD. Even so, the use of outdated network profiles and attack patterns may limit the validation of the proposed system. Also, the lack of a thresholding scheme specification for such a system may hinder its deployment in real environments.

Qu et al. [42] proposed a hybrid intrusion detection system for fog networks combining a vanilla Generative Adversarial Network and Long Short-Term Memory. The GAN model is trained on legitimate data to learn to generate normal samples representing the network behavior baseline. The LSTM model implements a regression algorithm for predicting a sample at time t based on previous data. The predicted sample is compared against the generated baseline, and instances exceeding a threshold are considered anomalous. A thorough set of experiments on the NSL-KDD dataset shows outperforming scores in quantitative metrics compared to state-of-the-art alternatives. However, the reliance on a single outdated dataset for experimental validation partially supports the system's generalizability. The IDS additionally misses a well-defined method for estimating the threshold.

Cheng et al. [43] discussed a low-cost anomaly detection system for intelligent IoT networks by applying Attention GAN. The generative model is compressed using knowledge distillation to reduce computational complexity, meeting the limited resources of intelligent IoT devices. Intrusion identification is carried out through data reconstruction and discrimination anomaly scores. The solution is tested on SWMRU, KDDCup99, and HomeC, achieving F1-scores above 90% while keeping compression performance degradation around 2%. Despite training with unlabeled data, the system still requires labeled data to validate and configure the hyperparameters besides not defining the applied inference threshold.

Wu et al. [44] introduced an intrusion detection algorithm for edge-based IoT networks. First, the optimal traffic features are selected using fuzzy logic. A trained CNN model is then applied to implement a Deep Convolutional GAN for classifying multiple cyber attacks. They used CIC-DDOS2019 and CSE-CIC-IDS2018 public benchmarks for evaluation, comparing with deep autoencoder, stacked autoencoder, and SVM. Despite overperforming its competitors, the proposed IDS may experience elevated labor in deployment and overtime false negative rate increase given its reliance on a limited set of labeled data.

Abdalgawad et al. [45] studied the applicability of generative models to IoT anomaly detection on the IoT-23 dataset. The Adversarial Autoencoder and Bidirectional GAN were implemented for supervised identification of attacks. The proposed networks are compared against baseline models, outperforming them in the experiments. While their experimental analysis has proven the classification capacity of such models, the system's suitability to real-world IoT networks is limited by the viability of recurrent data collection and labeling.

Anusha et al. [46] discussed a novel solution for intrusion detection on wireless sensor networks. The system integrates a dual interactive WGAN with war strategy optimization for parameter tuning. The discriminative network executes the supervised classification of samples through anomaly scoring and thresholding. The IDS reaches improved performance on NSL-KDD and CIC-IDS2017 benchmark datasets. Though, relying on labeled data for training and validating the model increases the system's deployment cost. The method for calculating the decision threshold is not discussed as well.

The scientific community has been investigating different generative models for intrusion detection, as summarized in Table I. We identified limitations requiring further discussion and examination. Multiple works applied outdated benchmark datasets such as KDD99 to evaluate their system, which may limit their validation experiments. Many authors rely on labeled data to set up their NIDS, presenting the unsupervised paradigm as a future research direction. Most reviewed works did not specify their applied thresholding method, a fundamental component for deploying such systems in real environments. The related works commonly do not incorporate a mitigation algorithm for constructing a fully automated defense system.

In this research, we tackle such issues by proposing an entropy-based f-AnoGAN model for anomaly detection in an SDN environment. The trained generative network calculates

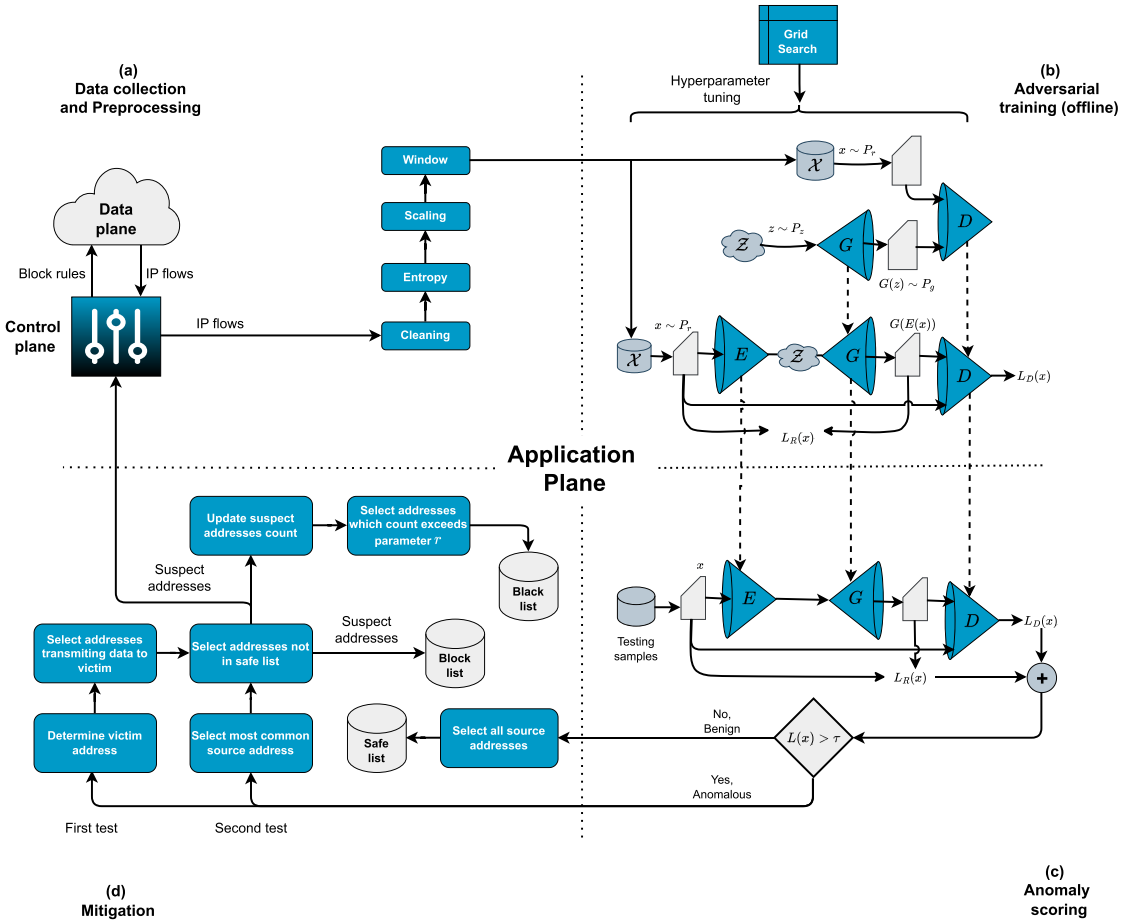

Fig. 1. Proposed system architecture: (a) IP flow data collection and preprocessing; (b) WGAN-GP and Encoder offline training and hyperparameter tuning; (c) Anomaly scoring and thresholding; (d) Mitigation.

anomaly scores for incoming traffic through efficient data encoding and decoding. A well-defined thresholding scheme promotes the system's straightforward installation on network environments. Our solution is also not dependent on labeled data in any configuration stage, reducing deployment overhead in real-world scenarios. We include a mitigation algorithm for countering arising volume anomalies without human intervention and keeping the network at regular transmission rates. The system is validated on four updated public datasets and compared against state-of-the-art models, presenting promising results.

IV. Proposed System

This section thoroughly presents the proposed anomaly detection and mitigation solution. It is designed for SDN environments but can be straightforwardly adapted to operate in generic TCP/IP networks that support IP flow data collection and exportation, as illustrated in Fig. 1. The security system is installed on the SDN's application plane and periodically communicates with the network controller to gather traffic data for analysis. Our system collects and analyzes IP flow data in one-second intervals, providing near real-time response to emerging traffic volume attacks. The first module preprocesses the incoming IP flows, which are submitted to an anomaly scoring analysis. The mitigation module identifies the suspicious addresses in anomalous traffic, which are then forwarded back to the network controller for blockage. The following subsections dive deeper into the system's configuration and operation considerations.

A. Data Preprocessing

The collected traffic is prepared before being sent to the anomaly detection algorithm, as presented in Fig. 1(a). The preparation includes cleaning, entropy calculus, scaling, and sliding window formatting.

1) Cleaning: Machine learning-based systems' efficacy depends on their data quality and integrity [47]. Ensuring the collected IP flow data is correct and complete is an important preprocessing step for a network anomaly detection system. Common causes for network data corruption are erroneous collection, hardware failure, and software bugs. Therefore, the incoming network samples must pass through an analysis to identify missing and corrupted values. For numeric IP flow features, we impute invalid values, such as empty field, infinity, or NaN (Not a Number), with a zero value, based on the study discussed by Mall et al. [48]. IP flows containing invalid qualitative features, including IP addresses and port numbers, are dropped since the proposed anomaly detection algorithm relies on these features for conducting volume anomaly detection.

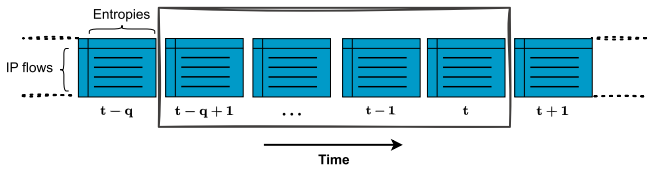

Fig. 2. Sliding window scheme.

2) Entropy: Volume anomalies can alter the usual probability distribution of traffic in various ways. For instance, Distributed Denial of Service (DDoS) attacks aim to tire out the victim's networking resources by suddenly increasing the latter's traffic exchange levels. The target device may quickly become the most common traffic destination in that scenario, unbalancing the overall distribution of the destination IP address feature. Scanning attacks iterate on various ports to gather information and discover vulnerabilities in the victim's services. The number of unique destination ports will likely increase during such a traffic anomaly, uniformizing the distribution of this feature [49], [50].

Information theory provides a mathematical tool that can be applied to quantify change in the distribution of traffic features: the Shannon Entropy [51]. This tool measures the uncertainty in a probability distribution for a certain random variable x. Like in the DDoS example, when the distribution is skewed towards some specific states, there is less uncertainty regarding the possible outcomes, thus reducing the entropy. Conversely, as in the scanning example, the closer the distribution is to uniform, the greater the uncertainty regarding the variable's outcome, increasing the entropy. The Shannon Entropy equation is defined in (4). $p(x_i)$ represents the probability of the i-th possible outcome for variable x. The equation then can be interpreted as the mean negative logarithm of the probability of the outcomes of x.

$$H(x) = -\sum_{i=1}^{n} p(x_i)\log_2[p(x_i)] \tag{4}$$

The collected IP flows are summarized into a sample comprising the individual entropies for source IP, destination IP, source port, and destination ports. This entropy sample represents the measured uncertainty for the distributions of addresses and ports for the analysis interval.

3) Scaling: The calculated entropies are scaled into the same numerical interval. Scaling is a standard preprocessing step when preparing data for machine learning algorithms since it reduces training time and facilitates models' convergence [9], [52]. Different problems solved using machine learning models may benefit from applying distinct data scaling techniques. The most effective scaling method for a target problem is commonly found through heuristics and experimentation. Based on the empirical results discussed by Radford et al. [53], we scale the entropy variables to fall within the range of -1 and 1.

4) Sliding Window: The proposed system tracks the entropy samples calculated in the last q seconds, including the one in the current timestamp t. These are concatenated to form a sliding window to provide a broader analysis context for the upcoming anomaly detection algorithm, as depicted in Fig. 2. When new traffic data is collected and transformed into an entropy sample, the latter is inserted into the sliding window. Since the window size is fixed to q and should have a time-sliding behavior, the insertion follows a first-in, first-out policy: the oldest sample $t - q + 1$ is removed, while the newly calculated one $t + 1$ is added to the window. The optimal size q for the sliding window is a hyperparameter that must be tuned to ensure anomaly detection effectiveness. In Sections IV-D and V-B, we discuss the procedure for adjusting q and its selected value, respectively.

B. Adversarial Training

Schlegl et al. [54] proposed the f-AnoGAN, an unsupervised generative model devised for image anomaly detection. In this work, we adapt the original proposal for volume anomaly detection in software-defined environments based on the calculated entropy features.

The f-AnoGAN model comprises two internal components trained separately: a Wasserstein GAN with gradient penalty and an encoder network, as depicted in Fig. 1(b). First, the WGAN-GP's generator G and discriminator D are trained on normal data to capture their probability distribution P_r. Its training objective is to minimize the Wasserstein distance between the generated data distribution P_g and the training data distribution P_r. After training, the generator network represents a function $G : z \to x$. G maps a latent vector z from l-dimensional latent space $\mathcal{Z} \in \mathbb{R}^l$ to a sample x in the m-dimensional training data space $\mathcal{X} \in \mathbb{R}^m$.

Subsequently, the encoder component E is trained on normal data to learn the generator's inverse transformation. The encoder is a function $E : x \to z$, which maps samples from data space $\mathcal{X}$ to latent space $\mathcal{Z}$. The encoding network training is conducted in an autoencoder setup, using the trained generator with fixed parameters as a decoding network. The training loss function is defined in (5), (6), and (7). During training, the encoder parameters are adjusted using stochastic gradient descent to minimize its loss $L(x)$. The $L_R(x)$ term represents the mean squared error between the incoming sample x and its reconstruction $G(E(x))$, where m is the dimensionality of x. The $L_D(x)$ term calculates the mean squared error on the discriminative high-level features of the sample and its respective reconstruction, weighted by a ϵ factor. The features are obtained by evaluating a function f on these samples. f corresponds to the n neuron activations in the last hidden layer from the WGAN-GP's trained discriminator, which precedes the output layer. The discriminator parameters are also kept unchanged throughout the encoder training.

$$L(x) = L_R(x) + L_D(x) \tag{5}$$

$$L_R(x) = \frac{1}{m}\sum_{i=1}^{m}(x - G(E(x)))^2 \tag{6}$$

$$L_D(x) = \frac{\epsilon}{n}\sum_{j=1}^{n}(f(x) - f(G(E(x))))^2 \tag{7}$$

After offline training completion, the encoding network learns to encode input samples fitting the training data distribution P_r. Since the encoder, generator, and discriminator are all trained on normal data, they are expected to present increased

residuals when reconstructing anomalous data outside P_r. In subsection IV-C, we describe how this behavior can be leveraged for online network anomaly detection.

C. Anomaly Scoring

When deployed in a real-world network, our f-AnoGAN-based system attributes an anomaly score to each analyzed network traffic window, representing their respective degree of anomalousness. Fig. 1(c) illustrates the anomaly scoring module. The score is calculated using the same loss function $L(x)$ applied in the encoder offline training, as introduced in (5). Since the f-AnoGAN is trained using only legitimate traffic data, the encoder is expected to compress anomalous samples poorly. Consequently, the generator network may reconstruct the respective compression faultily. The resulting difference between the original and reconstructed sample features may cause an increase in the calculated anomaly score for the corresponding traffic sample under analysis. The system labels an incoming traffic window anomalous whenever its anomaly score surpasses a predefined threshold τ. This limiting value represents the maximum degree of anomalousness the system tolerates.

The respective threshold is calculated by analyzing the probability distribution P_s of anomaly scores for normal window samples. It is well-known that approximately 95% of samples from a Gaussian distribution stay within two standard deviations from the mean. In this case, samples beyond that interval may be regarded as outliers. Since the distribution P_s is unknown, we cannot use the same rationale to define a threshold for identifying outliers. Therefore, we apply Chebyshev's theorem, as defined in (8), to estimate the spreading of samples around the mean [55]. X is the set of normal window anomaly scores drawn from P_s, μ symbolize the mean value, σ represents the standard deviation, and k indicates a distance factor from the mean. The theorem states that the probability of finding samples within k standard deviations from the mean is greater than or equal to $1 - \frac{1}{k^2}$. In this work, we want to ensure that at least 95% of the samples are within that interval around the mean. Solving (8) for k to determine its value given the desired confidence probability P results in (9). The value for k, which corresponds to a $P = 95\%$, is ≈ 4.47. Finally, the tolerance threshold τ is defined as the mean μ plus 4.47 times the standard deviation σ of the anomaly scores for the normal window samples, as given by (10). The selected value of k guarantees that at least 95% of scores for normal samples are within k standard deviations from the mean, and we may infer greater scores as representing outliers.

$$P(|X - \mu| \leq k\sigma) \geq \left(1 - \frac{1}{k^2}\right) \tag{8}$$

$$k = \sqrt{\frac{1}{1 - P}} \tag{9}$$

$$\tau = \mu(L(x)) + 4.47 \times \sigma(L(x)) \tag{10}$$

D. Hyperparameter Tuning

The hyperparameters represent determining variables to promote neural network training quality and model generalization.

TABLE II
F-ANOGAN'S HYPERPARAMETERS

Hyperparameter	Search space
window size (q)	2, 8, 16
latent space dimensions (l)	32, 64, 128
generator layers	1, 2, 3
generator neurons	8, 16, 32
discriminator layers	1, 2, 3
discriminator neurons	8, 16, 32
batch size	32, 64, 128
learning rate	1e−3, 1e−4, 1e−5

They can not be learned from data and must be tuned using a separate validation dataset during training. We apply a grid search algorithm during f-AnoGAN offline training to select appropriate values for its hyperparameters. Table II presents the search space defined for each regarded hyperparameter. The window size q corresponds to the number of $q - 1$ past seconds and the current second for the anomaly detection analysis. The l is the dimensionality for the encoder output and generator input. The generator and discriminator layers and neurons correspond to the architecture of their hidden structure, not considering the input and output layers. Note that the search space defined in Table II is not meant to be immutable. Network administrators may extend the search according to their needs, available data, and computing resources. For example, the search could include more complex generator and discriminator architectures to meet scalability demands when the target network produces a greater traffic volume.

The searching algorithm instantiates and trains the f-AnoGAN model using each possible value combination for the set of hyperparameters. Since there are 8 hyperparameters with search spaces of size 3, the total amount of possible combinations is $3^8 = 6561$. Each evaluated hyperparameter set is ranked according to the validation score v it yields for the f-AnoGAN model during training. v is the median plus the median absolute deviation (MAD) of legitimate samples' anomaly scores $L(x)$, as defined in (11). The closer the median and MAD values are to 0, the less the model error when reconstructing normal traffic samples. In that case, the hyperparameter set with a v score close to 0 enables the model to encode training samples properly. At the end of the search, the hyperparameter set that provides the minimum validation score is selected as the final configuration for the f-AnoGAN model. We use median-based metrics to define the validation score instead of mean-based ones since the latter could be biased towards the anomaly score of any outlying legitimate sample.

$$v = Median(L(x)) + MAD(L(x)) \tag{11}$$

We manually adjusted some other hyperparameters not included in the grid search, considering recent reports published by the scientific community. Related works have previously tuned their values based on heuristics and empirical evidence.

Schlegl et al. [54] point out estimations for some hyperparameter values of the f-AnoGAN. First, they model the latent space using the normal distribution with a mean of 0 and a

standard deviation of 1. The activation for the output layer of the encoder network is set as the hyperbolic tangent, restricting the values of the codified features in the range of -1 and 1. The WGAN-GP's parameters are optimized using Adam, while the encoder ones are adjusted utilizing RMSprop. During the WGAN-GP training, the discriminator's parameters are updated five times for each update in the generator. The discriminative network also contains layer normalization for more stable training. Since the encoder is trained in an autoencoder setup, its layering architecture equals the generator's in reverse order: the latter last layer is the former first layer, and so on. The authors assign a value of 1 for the ϵ weighting factor applied in both the encoder loss and the anomaly score function, meaning that the reconstruction error and the feature error have the same impact on the function output.

Radford et al. [53] reported suggestions for stabilizing generative networks training, such as data scaling in the interval of -1 and 1, and consequently, generator output activation as the hyperbolic tangent function to match these values. Their model further applies the LeakyReLu activation function with a 0.2 slope for network hidden layers. Finally, considering the promising results acquired in recent works [56], [57], we set 20% dropout regularization layers in the discriminator network to avoid overfitting and encourage model generalization.

E. Mitigation

The mitigation module is invoked for every second of analysis after the anomaly scoring execution. Its executed action is conditioned on the classification attributed to the traffic window under analysis, as presented in Fig. 1(d). If the data is inferred as legitimate, all the IP addresses in the network traffic for the last second are saved to a so-called safe list data structure. The devices whose addresses are inserted on this list can freely exchange traffic in the network for the next p seconds. When their communication time is over, they must show legitimate behavior by participating in benign connections to be added to the safe list again.

If the incoming traffic data is inferred as anomalous, the mitigation module executes a subroutine to identify the IP addresses probably responsible for the abnormal increase in the traffic volume. The module conducts two tests to recognize these suspicious addresses. The first one determines the potential victim device, represented by the destination IP address receiving the most traffic. All source addresses communicating with the victim are classified as suspicious. The second test resolves the most frequent source address in IP flow traffic. The suspect addresses not in the safe list are then added to a blocking list for p seconds. The safe list prevents blocking legitimate addresses that send harmless traffic to the victim. All addresses on the blocking list at the current analysis time are reported to the SDN controller. The latter creates blocking rules and installs them on the data plane devices to mitigate traffic from untrustworthy sources.

After experimentation, we set the time parameter p as a random integer between 1 and 10 for every new address insertion to any list. The safe and blocking lists are fundamental to reducing the impact of the anomaly detection algorithm's wrong inferences. A false positive traffic classification may lead to a temporary blockage of legitimate devices for p seconds. In case of false negative inferences, malicious users may freely generate anomalous traffic during the same period.

A third data structure referred to as a black list is responsible for keeping track of addresses that recurrently are labeled as suspects. Throughout the NIDS execution, the mitigation module counts each time a unique address is flagged as suspicious and added to the blocking list. Addresses classified as suspects more than r times are permanently inserted into the black list due to their recurrent presence in anomalous events. Using a black list aids in gradually capturing the addresses used by malicious users to launch the attacks. The SDN controller always blocks black-listed addresses independently from the safe list state. We conducted experiments to adjust the hyperparameter r, setting it to 500, 1000, 1500, and 2000 values. The selected value, which generalizes to all experimental scenarios, is 1000.

F. Complexity Analysis

Complexity analysis is a fundamental tool in computer science. It studies and predicts algorithms' behavior when processing increasing input sizes, a common scenario in real-world applications. In this subsection, we evaluate the computational complexity of the proposed system for its online execution. The collected IP flows must be temporarily stored and iterated during the data preprocessing stage to measure the address and port entropies. The spatial and temporal complexity presented by this procedure is $O(n)$, where n is the number of IP flows. The anomaly detection algorithm analyzes a sliding window of entropy samples. Since it is not directly dependent on the amount of IP flows, its computational complexity is $O(1)$. The mitigation algorithm execution relies on storing the IP flows, processing each individually, and accessing lists. The safe, block, and black lists are implemented as dictionary data structures, thus having unchanging access time. Therefore, the computational complexity observed in the mitigation module is $O(n)$.

V. Experiments

In this Section, we present different network scenarios to experimentally validate the proposed system's anomaly detection and mitigation performance. These scenarios are represented by IP flow datasets commonly used by researchers to benchmark intrusion detection systems. The system's online processing pipeline (Fig. 1) is executed offline utilizing the datasets' static traffic. Firstly, the system is constructed to collect and analyze traffic in one-second intervals. Therefore, for each studied dataset, we group the IP flows according to the starting second found in their timestamp feature. Secondly, the proposed system's interaction with the control plane for data collection is reproduced by iteratively reading each IP flow group from the dataset under experimentation. Finally, the system analyzes IP flow groups by conducting data preprocessing, adversarial training, anomaly detection, and mitigation, as discussed in Section IV. The validation experiments are run on a computer equipped with an AMD Ryzen 7 5700 U with 12 GB of RAM. The applied software includes Ubuntu 22.04.4 LTS, Python 3.10.9, Tensorflow 2.11.0, Keras 2.11.0, and Scikit-learn 1.2.1.

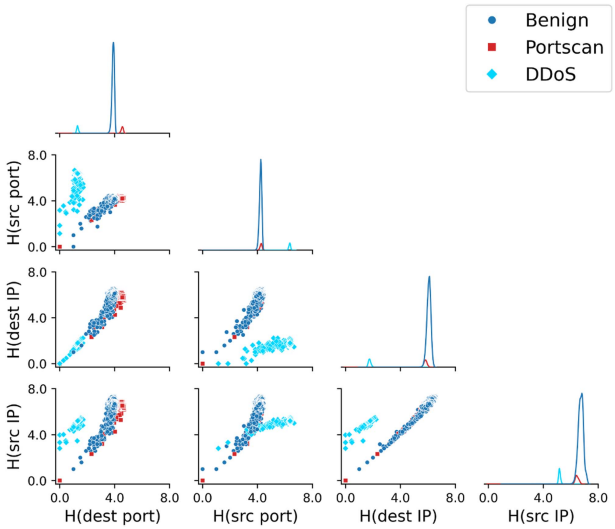

Fig. 3. Entropies of addresses and ports for the Orion test set.

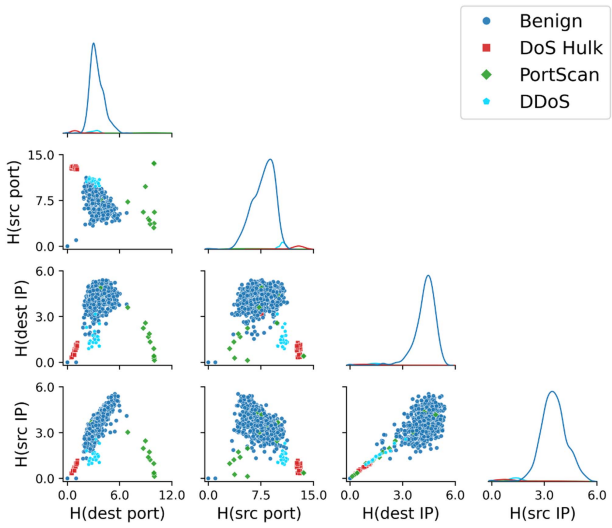

Fig. 5. Entropies of addresses and ports for the CIC-IDS2017 test set.

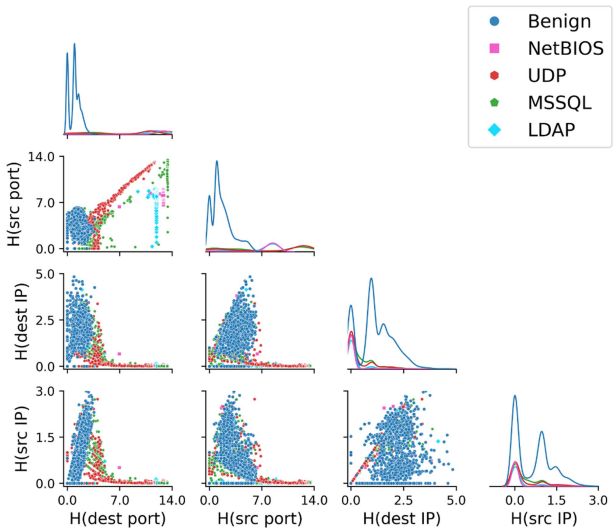

Fig. 4. Entropies of addresses and ports for the CIC-DDoS2019 test set.

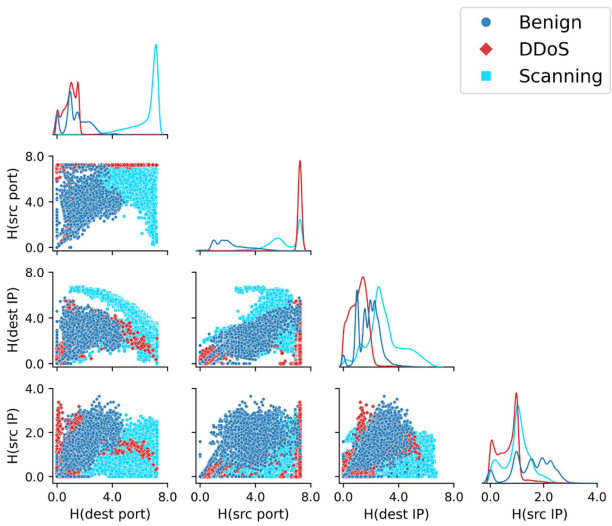

Fig. 6. Entropies of addresses and ports for the TON_IoT test set.

A. Datasets

Four public datasets—Orion, CIC-DDoS2019, CIC-IDS-2017, and TON_IoT—represent the experimental network scenarios. These datasets provide distinct legitimate traffic profiles and updated volume anomalies (e.g., DDoS, PortScan), which are essential for evaluating our proposal. Recent literature reviews show that the CIC datasets are among the most used data sources for anomaly detection benchmarking [20], [58].

The IP flow data for the datasets' test sets were grouped by timestamp and preprocessed following the method described in subsection IV-A. The data preparation results in four new traffic variables: source IP address entropy, destination IP address entropy, source port entropy, and destination port entropy. Figs. 3, 4, 5, and 6 present a visualization of the pairwise relationships for the variables. There are six unique pairs of addresses and ports entropies. Each subgraph in the figures corresponds to a distinct pair.

1) Orion: The Orion dataset was generated by the Orion Research Group from the State University of Londrina [23]. The researchers applied the Mininet tool to emulate a Software-Defined Network with 6 switches and 140 hosts. The emulation ran for three days, yielding 72 hours of IP flow data. The first one comprises traffic from legitimate users, representing the normal profile of network data exchange. In the second and third days, the network is targeted by PortScanning and Distributed Denial of Services attacks during morning and afternoon activities. For conducting the experimental analysis, we consider the first day as the training set for configuring the proposed system and the third day as the test set for evaluating it. We preprocessed these sets using the approach discussed in subsection IV-A. Fig. 3 displays the possible pairs of entropies of addresses and ports for the test set.

2) CIC-DDoS2019: The CIC-DDoS2019 dataset was developed by the Canadian Institute of Cybersecurity (CIC) [24]. The proposed testbed is composed of 1 server, 1 firewall, 2 switches,

TABLE III
F-ANOGAN'S HYPERPARAMETER SEARCH RESULT

Parameter	Orion	CIC-DDoS2019	CIC-IDS2017	TON_IoT
window size	2	2	2	2
latent space dimensions	128	128	128	128
generator layers	1	2	1	2
generator neurons	32	32	32	32
discriminator layers	3	1	1	1
discriminator neurons	32	32	16	32
batch size	32	32	32	32
learning rate	1e−5	1e−5	1e−5	1e−5

TABLE IV
MODELS' TRAINING TIME IN SECONDS

Model	Orion	CIC-DDoS2019	CIC-IDS2017	TON_IoT
f-AnoGAN [56]	222.72	45.65	10.55	32.28
f-AnoGAN$_{izi}$	191.18	46.72	10.19	31.47
f-AnoGAN$_{f}$	205.35	44.31	9.36	31.63
Autoencoder	**16.0**	**8.05**	3.09	11.49
BiGAN [27]	31.85	10.99	**2.67**	**8.69**
FID-GAN [21]	239.09	69.78	12.54	49.22

and 4 personal computers. The IP flow traffic was captured on two different days, beginning in the morning around 9:00 AM and finishing in the late afternoon near 5:00 PM. On the first day, besides the regular traffic, the target network experiences anomalous traffic corresponding to multiple variations of the DDoS attack: NTP, DNS, LDAP, MSSQL, NetBIOS, SNMP, SSDP, UDP, UDP-Lag, WebDDoS, SYN, and TFTP. The second traffic day is also affected by PortMap, NetBIOS, LDAP, MSSQL, UDP, UDP-Lag, and SYN attacks. We removed the attacks from the first day to create a proper training dataset for our system. The second day was used as the testing set. The result of preprocessing the latter is illustrated in Fig. 4.

3) CIC-IDS2017: The CIC institution also makes the CIC-IDS2017 dataset available [25]. The implemented network has standard equipment such as a router, firewall, server, and personal computer. There were five traffic days, with the capture starting Monday and finishing Friday. The first day covers only regular users' activity, while the other contains traffic from many attack families, including volume-based ones, namely DoS, DDoS, and Portscan. The IP flows composing this dataset include solely the hour and minute fields in the timestamp feature, impeding our system from analyzing the traffic on a second basis. Larger collection intervals may hide brief anomalous patterns among legitimate traffic, potentially increasing the system's false negative rate. We still test the system in this scenario to evaluate how it reacts under such execution conditions. Monday is used for training our system, and Wednesday and Friday are used jointly as the test set. The preprocessed entropies for the test set are pairwisely plotted for visualization in Fig. 5.

4) Ton_Iot: The TON_IoT dataset was published in 2021 by researchers from the University of New South Wales (UNSW) [26]. They implemented an SDN and NFV-based environment to simulate contemporary real-world networks realistically. The proposed network testbed is built in a laboratory, comprising IoT, edge, fog, and cloud devices. The collected dataset contains IoT sensor measurements, operating system logs, and network traffic samples. The latter is available in the format of IP flows, which are considered for evaluating our proposed system. There are multiple families of attacks among realistic, legitimate traffic. We investigate the volume-based families in the following experiments, including DDoS and scanning. These attacks present distinct behavioral patterns compared to the ones in previous benchmarks. In Orion, CIC-DDoS2019, and CIC-IDS2017, the attacks are usually carried from multiple malicious sources to a single victim (i.e., n-to-1). Conversely, in TON_IoT, the attacks are commonly conducted from many sources to many targets (i.e., n-to-n). We split the raw IP flow traffic into training and testing sets besides applying the preprocessing presented in subsection IV-A. Attack events are removed from the training set, allowing our system to learn only the network's normal behavior. Fig. 6 illustrates the pair plot for the testing set's entropies of addresses and ports.

B. System Setup

The proposed system is separately configured and trained in the four scenarios. Table III presents the selected values for the f-AnoGAN hyperparameters. The model converged using the same window size, latent space dimensions, generator neurons, batch size, and learning rate in all datasets. The other hyperparameters also show similar values in the three scenarios. The similarity in the results suggests that the f-AnoGAN optimal configuration is stable and has low variation across distinct testing scenarios. We also trained two additional models, namely f-AnoGAN$_{izi}$ and f-AnoGAN$_{f}$, to conduct an ablation study on the f-AnoGAN model's training loss. We aim to investigate the individual importance of the terms in the encoder loss described in (5). The f-AnoGAN$_{izi}$ is trained using only the $L_R(x)$ term, while the f-AnoGAN$_{f}$ solely utilizes the $L_D(x)$ term.

We also implemented state-of-the-art neural networks, including Autoencoder, BiGAN [27], and FID-GAN [21], to compare performance with our system. We adapt them to calculate anomaly scores using reconstruction error based on sliding windows, as described in Section IV. To promote a fair comparison, we apply the f-AnoGAN's hyperparameter configuration to train them. These models were selected as a baseline for comparison, given their similarity to the proposed f-AnoGAN in terms of architecture and training procedure. They comprise encoder and decoder networks to compute sample reconstruction error on an unsupervised training setup, which supports their applicability in anomaly detection.

The training times taken by the implemented models are displayed in Table IV. Considering the stochastic behavior of learning algorithms, we trained each model three times and averaged the results. Orion, CIC-DDoS2019, CIC-IDS2017, and TON_IoT have 86400, 5524, 487, and 4899 traffic samples in their training sets. Since the Orion dataset has more samples than the CIC ones, the models usually take more time to train in the former. Given its simpler architecture and learning procedure, the autoencoder is the fastest model to train, taking 16.0 seconds in the worst-case scenario (Orion). The BiGAN model is faster than the other GAN-based models as it is implemented with regular dense layers, and the three internal networks are trained

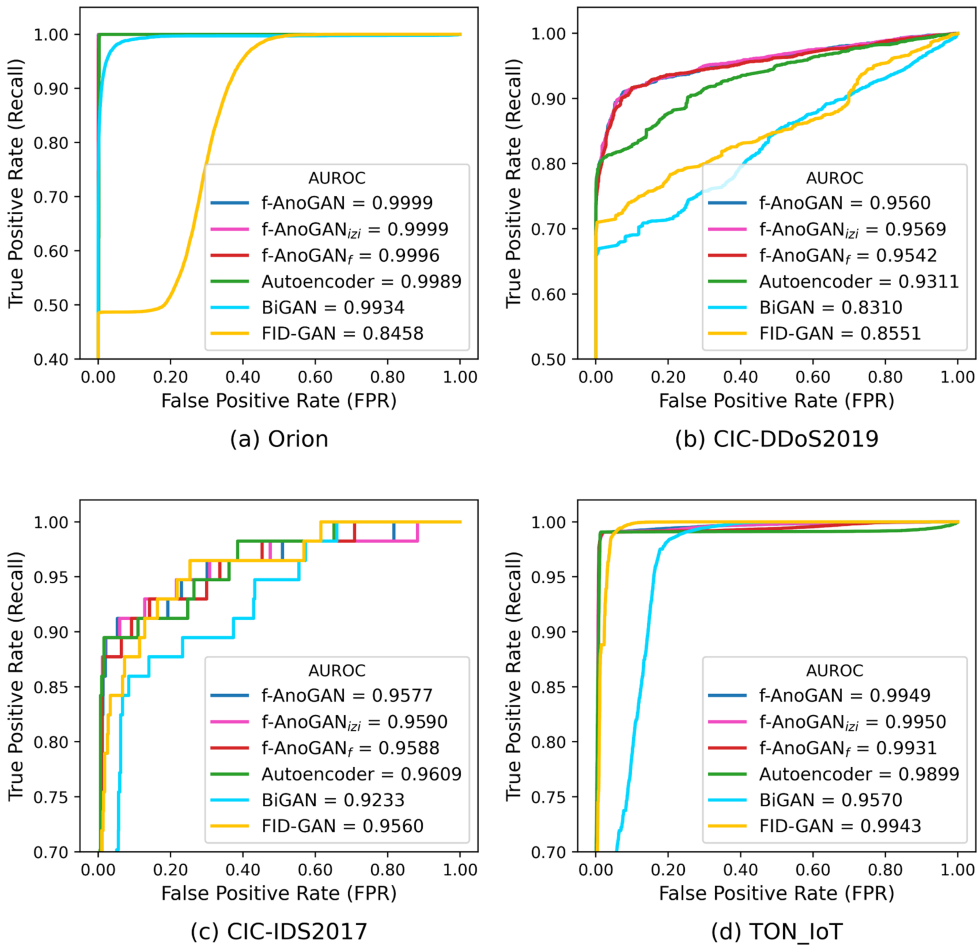

Fig. 7. ROC curves for the models.

in parallel. The f-AnoGAN-based models have similar training times when the dataset is small (CIC scenarios). In the worst case, the f-AnoGAN$_{izi}$ is faster than the others, requiring 191.18 seconds to train. The internal architecture based on LSTM cells increases the FID-GAN model's computational complexity, making it the slowest model to train, with 239.09 seconds in the worst-case scenario.

C. Detection

The anomaly detection task can be interpreted as a binary classification problem. Depending on its correctness, each inference produced by the detection system is regarded as a true positive, true negative, false positive, or false negative. The research community commonly summarizes the quality of these inferences using quantitative metrics, including the ROC curve, Area Under the ROC curve (AUROC), Precision, Recall, F1-score, and Matthews Correlation Coefficient (MCC) [59], [60], [61]. We evaluate the trained models on the test set of each scenario using such metrics.

Fig. 7 presents the ROC curve and the AUROC for the implemented models on the four test sets. The curve represents how well a model can correctly identify attacks while minimizing its false positive rate. The more the curve is to the upper left corner of the graph, the better the model can separate benign and malign samples, easing the definition of an arbitrary decision threshold. According to the ROC curves for the Orion scenario (a), all models can properly separate positive and negative classes except for the FID-GAN model, which has an AUROC of 0.8458. The curves for the f-AnoGAN models and the Autoencoder overlap in the upper left corner, and therefore, only the latter is visible. The f-AnoGAN$_{izi}$ model achieves the best AUROC in the CIC-DDoS2019 scenario (b), scoring 0.9569 points. The remaining f-AnoGAN models and the Autoencoder secure AUROC scores above 0.90. BiGAN and FID-GAN struggle to separate the classes with AUROCs of 0.8310 and 0.8551. In the CIC-IDS2017 scenario (c), all models obtain AUROCs above 0.90, suggesting efficiency regarding class separation. The Autoencoder and the f-AnoGAN$_{izi}$ models acquire the best scores of 0.9609 and 0.9590, respectively. Lastly, in the TON_IoT scenario (d), the f-AnoGAN variations and Autoencoder's curves again overlap in the upper left corner. The models present improved class separation capacity, with the f-AnoGAN$_{izi}$ achieving the peak AUROC of 0.9950.

Table V shows the implemented models' quantitative performance in anomaly detection in the Orion test set. All f-AnoGAN-based models acquire MCC and F1-scores above 0.98, underlining their effectiveness in unsupervised anomalous pattern identification. The remaining models achieved low Recall scores, which kept their MCC measure below 0.8. We point

TABLE V
ORION TESTING PERFORMANCE

Model	Precision	Recall	F1-score	MCC
f-AnoGAN [56]	0.9667	**1.0000**	0.9830	0.9811
f-AnoGAN$_{izi}$	0.9656	1.0000	0.9825	0.9805
f-AnoGAN$_{f}$	0.9758	0.9981	**0.9868**	**0.9853**
Autoencoder	0.9757	0.6525	0.7820	0.7801
BiGAN [27]	**0.9978**	0.4888	0.6561	0.6776
FID-GAN [21]	0.9051	0.4862	0.6326	0.6374

TABLE VI
CIC-DDOS2019 TESTING PERFORMANCE

Model	Precision	Recall	F1-score	MCC
f-AnoGAN [56]	0.9518	**0.8410**	**0.8930**	**0.8323**
f-AnoGAN$_{izi}$	0.9525	0.8404	0.8929	0.8323
f-AnoGAN$_{f}$	0.9598	0.8201	0.8845	0.8228
Autoencoder	0.9819	0.7995	0.8813	0.8250
BiGAN [27]	**1.0000**	0.6517	0.7891	0.7266
FID-GAN [21]	1.0000	0.6644	0.7984	0.7362

TABLE VII
CIC-IDS2017 TESTING PERFORMANCE

Model	Precision	Recall	F1-score	MCC
f-AnoGAN [56]	0.7777	**0.8596**	0.8166	0.8060
f-AnoGAN$_{izi}$	0.8305	0.8596	**0.8448**	**0.8353**
f-AnoGAN$_{f}$	0.8166	0.8596	0.8376	0.8277
Autoencoder	**0.8823**	0.7894	0.8333	0.8251
BiGAN [27]	0.8400	0.3684	0.5121	0.5406
FID-GAN [21]	0.8285	0.5087	0.6304	0.6336

TABLE VIII
TON_IOT TESTING PERFORMANCE

Model	Precision	Recall	F1-score	MCC
f-AnoGAN [56]	0.9924	**0.9905**	**0.9915**	**0.9691**
f-AnoGAN$_{izi}$	0.9924	0.9902	0.9913	0.9683
f-AnoGAN$_{f}$	0.9923	0.9904	0.9914	0.9687
Autoencoder	0.9981	0.8021	0.8894	0.7218
BiGAN [27]	**1.0000**	0.4589	0.6291	0.4339
FID-GAN [21]	1.0000	0.4191	0.5907	0.4060

out that the Autoencoder and BiGAN models could not perform well in the anomaly detection task despite effectively separating benign and malign classes with AUROCs surpassing 0.99 in this scenario. After investigation, we found that the drop in performance is explained by the applied unsupervised threshold scheme introduced in subsection IV-C. The selected threshold is suboptimal for these models. Their performance could be improved by searching for an optimal threshold value. This search, however, would require labeled attack data, losing the practicability of a fully unsupervised model. Conversely, the f-AnoGAN models are compatible with the unsupervised Chebyshev-based thresholding method, achieving high performance.

The detection performance of the models in the CIC-DDoS2019 test set is available in Table VI. Compared to the previous scenario, the f-AnoGAN models experienced a depletion in performance, having an MCC of approximately 0.83. Conversely, the Autoencoder, BiGAN, and FID-GAN networks presented improved MCCs of 0.8250, 0.7266, and 0.7362. The reduced MCC scores of all models are explained by their lower recall values, which are caused by the increase in false negative inferences. We found that these unidentified attack samples generally come from the start or end of the DDoS lunching intervals. In these specific moments, the anomalous traffic exchange rate is low and does not change the expected volume of regular traffic. The threshold suboptimality problem also contributes to the reduced performance in all models, having a lower impact in the f-AnoGAN-based models.

Table VII points out the anomaly detection quality for each model in the CIC-IDS2017 test set. The f-AnoGAN$_{izi}$ achieve the highest MCC of 0.8353. The remaining f-AnoGAN-based models and the Autoencoder obtain an MCC of around 0.82. The thresholding problem becomes more expressive in the BiGAN and FID-GAN models as they acquire low Recall and MCC scores.

The quantitative metrics for anomaly detection in TON_IoT test set are displayed in Table VIII. The Autoencoder model has suffered once more from the threshold suboptimality problem, as supported by its high AUROC metric and its reduced Recall value of 0.8021. The BiGAN and FID-GAN models perform worse than the previous three scenarios due to reaching Recall values below 0.5. The f-AnoGAN models acquire close performance with MCC values above 0.96.

Based on these results, we conclude that the f-AnoGAN$_{izi}$ model is the best for the following reasons. The ROC curves show that the f-AnoGAN$_{izi}$ has the highest AUROC in Orion, CIC-DDoS2019, and TON_IoT scenarios while staying only 0.0019 points below the best score in the CIC-DDoS2017. The f-AnoGAN-based models are generally less impacted by the threshold suboptimality problem, discarding the need for precise threshold searching based on labeled data. Consequently, the f-AnoGAN$_{izi}$ achieves the best MCC score in the CIC scenarios, keeping close to the highest score in the Orion and TON_IoT scenarios. Finally, among the f-AnoGAN models, the izi implementation has the lowest training time in the worst-case scenario (Orion). The explanation for being faster is that the model is not dependent on the discriminator network for calculating its loss as the other two variations, reducing the number of required calculations.

D. Mitigation

We now evaluate the security system's IP flow discarding efficiency. The anomaly scoring and thresholding module based on the f-AnoGAN$_{izi}$ implementation is executed cooperatively with the mitigation module, as depicted in Fig. 1. To conduct the investigation, for each experimental scenario, we set a vector containing the system's inferred classifications for the complete test set. A vector element represents the produced label for a single second of traffic. We iterate the vector feeding each element and corresponding IP flow traffic to the mitigation module, executing the procedure described in the subsection IV-E. After the iterations, we account for the total number of forwarded and dropped IP flows.

Table IX presents the forwarded and dropped IP flows for the Orion test set. The system can drop 99% of malign traffic,

TABLE IX
IP FLOW MITIGATION ON ORION'S TEST SET

Flow class	Forwarded	Blocked
Benign	9,414,550 (97%)	303,752 (3%)
Malign	553 (1%)	2,159,738 (99%)

TABLE X
IP FLOW MITIGATION ON CIC-DDOS2019'S TEST SET

Flow class	Forwarded	Blocked
Benign	55,612 (98%)	1353 (2%)
Malign	4,166 (1%)	15,223,061 (99%)

TABLE XI
IP FLOW MITIGATION ON CIC-IDS2017'S TEST SET

Flow class	Forwarded	Blocked
Benign	852,492 (99%)	3,859 (1%)
Malign	10,892 (2%)	506,621 (98%)

TABLE XII
IP FLOW MITIGATION ON TON_IOT'S TEST SET

Flow class	Forwarded	Blocked
Benign	371,534 (95%)	21,438 (5%)
Malign	315,877 (5%)	6,333,408 (95%)

restoring the network's normal traffic volume. Most benign IP flows are forwarded, indicating a low impact on legitimate users. A small portion of 3% is discarded, given the false positive inferences on the anomaly identification step.

The mitigation module's forwarding and traffic blockage for the CIC-DDoS2019 scenario are shown in Table X. Despite a reduced MCC of 0.8323 in the detection step, the system could block approximately 99% of malicious IP flows. The mitigation module's block list proved essential to keep up with the detection module's false negatives and block the malign IP addresses. The dropping had almost no impact on benign traffic since only 2% of real traffic was blocked.

Table XI displays the quantitative performance of the mitigation module on the CIC-IDS2017 dataset. Once again, the block list helps the mitigation algorithm to correctly capture malicious traffic despite the reduced Recall in the detection stage. The anomaly detection MCC of 0.8353 did not impede the system of discarding 98% of malign IP flows. Also, legitimate users are barely impacted as 99% of benign traffic is correctly forwarded.

The resulting drop performance for the proposed system in the TON_IoT test set is arranged in Table XII. The mitigation algorithm can block 95% of anomalous flows while also forwarding 95% of benign flows. The TON_IoT represents a more challenging attack scenario compared to the previous ones. Multiple malicious computers target multiple victim hosts, distributing the flood of traffic across many targets. During experimentation, we found that the safe and block list schemes struggle to identify all malign sources. The mitigation module, however, successfully applied the black listing scheme to identify the majority of attack sources over time. Further study of mitigation strategies for such abnormal attack settings (i.e., many sources to many targets) remains a future research direction.

VI. CONCLUSION

This research discussed a novel unsupervised volume anomaly detection and mitigation system for software-defined networks. We trained an entropy-based f-AnoGAN to model low-dimensional legitimate traffic, acquiring high detection rates while reducing computational overhead. The generative network calculates reconstruction residuals from incoming data samples, which are inferred anomalous if a threshold value is surpassed. The mitigation module is set to identify the suspicious addresses participating in anomalous events, which are subsequently blocked by the network controller. The proposed system is independent of labeled data for training and hyperparameter tuning, enabling straightforward installation in real networks.

We examine the proposed system's validity by running experiments on four public datasets: Orion, CIC-DDoS2019, CIC-IDS2017, and TON_IoT. The system hyperparameters are tuned using a grid search approach. Similar state-of-the-art models, namely Autoencoder, BiGAN, and FID-GAN, are implemented for comparison purposes. We also implemented two variations of the f-AnoGAN model to conduct an ablation study, which we refer to as f-AnoGAN$_{izi}$ and f-AnoGAN$_{f}$. Each model is trained on a subpart of the original f-AnoGAN's loss to assess their respective impact on anomaly detection performance.

The experimental results suggest that the f-AnoGAN$_{izi}$ variation is better than the complete f-AnoGAN and the state-of-the-art comparison models. It presents high AUROC values in the experimental scenarios. Differently from similar models, the izi variation is well adapted to the unsupervised thresholding scheme based on Chebyshev's theorem. Consequently, the model can leverage its class separation capabilities and achieve the highest MCC scores in anomaly detection. The f-AnoGAN$_{izi}$ is faster to train in the worst-case scenario than the f-AnoGAN and f-AnoGAN$_{f}$. These findings show that the complete f-AnoGAN, as introduced by Schlegl et al. [54], may not be the best option in different application contexts.

We run the izi detection model cooperatively with the mitigation module to assess the proposed system's efficiency regarding malicious traffic dropping. Despite suffering from false negative inferences, the system could block between 95% and 99% of malign IP flows in the experimental scenarios while forwarding the most benign traffic. Even with the potential blockage of legitimate users due to false positives, they can soon communicate in the network again since they are only blocked by a finite time. Therefore, the system offers a positive cost-benefit as it trades off a low false positive rate for effectively countering volume anomalies.

In our future work, we plan to experiment with more benchmark datasets to evaluate the system in different scenarios. Thresholding robustness is pivotal towards efficacy in unsupervised anomaly detection and mitigation. We aim to study alternative threshold definition methods as well as mitigation algorithms, exploring their generalization to distinct network scenarios. We intend to integrate explainable artificial intelligence into the proposed system, promoting understanding, reliance, and simpler debugging.

REFERENCES

[1] L. R. Frank, A. Galletta, L. Carnevale, A. B. Vieira, and E. F. Silva, "Intelligent resource allocation in wireless networks: Predictive models for efficient access point management," *Comput. Netw.*, vol. 254, 2024, Art. no. 110762, doi: 10.1016/j.comnet.2024.110762.

[2] D. Li, "Virtualization and energy management optimization of high speed computer network data centers based on optical switching and network technology," *Thermal Sci. Eng. Prog.*, vol. 55, 2024, Art. no. 102918, doi: 10.1016/j.tsep.2024.102918.

[3] A. Baz, J. Logeshwaran, Y. Natarajan, and S. K. Patel, "Enhancing mobility management in 5G networks using deep residual LSTM model," *Appl. Soft Comput.*, vol. 165, 2024, Art. no. 112103, doi: 10.1016/j.asoc.2024.112103.

[4] A. Rego, L. Garcia, S. Sendra, and J. Lloret, "Software defined network-based control system for an efficient traffic management for emergency situations in smart cities," *Future Gener. Comput. Syst.*, vol. 88, pp. 243–253, 2018, doi: 10.1016/j.future.2018.05.054.

[5] D. M. B. Lent, M. P. Novaes, L. F. Carvalho, J. Lloret, J. J. Rodrigues, and M. L. Proença, "A gated recurrent unit deep learning model to detect and mitigate distributed denial of service and portscan attacks," *IEEE Access*, vol. 10, pp. 73229–73242, 2022, doi: 10.1109/ACCESS.2022.3190008.

[6] A. A. Wabi, I. Idris, O. M. Olaniyi, and J. A. Ojeniyi, "DDOS attack detection in SDN: Method of attacks, detection techniques, challenges and research gaps," *Comput. Secur.*, vol. 139, 2023, Art. no. 103652, doi: 10.1016/j.cose.2023.103652.

[7] N. S. Musa, N. M. Mirza, S. H. Rafique, A. Abdallah, and T. Murugan, "Machine learning and deep learning techniques for distributed denial of service anomaly detection in software defined networks-current research solutions," *IEEE Access*, vol. 12, pp. 17982–18011, 2024, doi: 10.1109/ACCESS.2024.3360868.

[8] S. Sendra, A. Rego, J. Lloret, J. M. Jimenez, and O. Romero, "Including artificial intelligence in a routing protocol using software defined networks," in *Proc. 2017 IEEE Int. Conf. Commun. Workshops*, 2017, pp. 670–674, doi: 10.1109/ICCW.2017.7962735.

[9] Z. Li, C. Huang, and W. Qiu, "An intrusion detection method combining variational auto-encoder and generative adversarial networks," *Comput. Netw.*, vol. 253, 2024, Art. no. 110724, doi: 10.1016/j.comnet.2024.110724.

[10] L. Nie et al., "Intrusion detection for secure social Internet of Things based on collaborative edge computing: A generative adversarial network-based approach," *IEEE Trans. Computat. Social Syst.*, vol. 9, no. 1, pp. 134–145, Feb. 2022, doi: 10.1109/TCSS.2021.3063538.

[11] T. Kim and W. Pak, "Early detection of network intrusions using a GAN-based one-class classifier," *IEEE Access*, vol. 10, pp. 119357–119367, 2022, doi: 10.1109/ACCESS.2022.3221400.

[12] Z. Liu, J. Hu, Y. Liu, K. Roy, X. Yuan, and J. Xu, "Anomaly-based intrusion on IoT networks using AIGAN-a generative adversarial network," *IEEE Access*, vol. 11, pp. 91116–91132, 2023, doi: 10.1109/ACCESS.2023.3307463.

[13] J. Zhang, Z. Pan, J. Cui, H. Zhong, J. Li, and D. He, "Toward open-set intrusion detection in VANETs: An efficient meta-recognition approach," *IEEE Trans. Netw. Sci. Eng.*, vol. 11, no. 6, pp. 6589–6604, Nov./Dec. 2024, doi: 10.1109/TNSE.2024.3459087.

[14] J. Xu, X. Li, P. Wang, X. Jin, and S. Yao, "Multi-modal noise-robust DDoS attack detection architecture in large-scale networks based on tensor SVD," *IEEE Trans. Netw. Sci. Eng.*, vol. 10, no. 1, pp. 152–165, Jan./Feb. 2023, doi: 10.1109/TNSE.2022.3205708.

[15] D. S. Mary, L. J. S. Dhas, A. Deepa, M. A. Chaurasia, and C. J. J. Sheela, "Network intrusion detection: An optimized deep learning approach using Big Data analytics," *Expert Syst. With Appl.*, vol. 251, 2024, Art. no. 123919, doi: 10.1016/j.eswa.2024.123919.

[16] W. Lim, K. Y. S. Chek, L. B. Theng, and C. T. C. Lin, "Future of generative adversarial networks (GAN) for anomaly detection in network security: A review," *Comput. Secur.*, vol. 139, 2024, Art. no. 103733, doi: 10.1016/j.cose.2024.103733.

[17] T. Talaei Khoei and N. Kaabouch, "A comparative analysis of supervised and unsupervised models for detecting attacks on the intrusion detection systems," *Information*, vol. 14, no. 2, 2023, Art. no. 103, doi: 10.3390/info14020103.

[18] T. K. Boppana and P. Bagade, "GAN-AE: An unsupervised intrusion detection system for MQTT networks," *Eng. Appl. Artif. Intell.*, vol. 119, 2023, Art. no. 105805, doi: 10.1016/j.engappai.2022.105805.

[19] G. F. Scaranti, L. F. Carvalho, S. B. Junior, J. Lloret, and M. L. Proença Jr, "Unsupervised online anomaly detection in software defined network environments," *Expert Syst. With Appl.*, vol. 191, 2022, Art. no. 116225, doi: 10.1016/j.eswa.2021.116225.

[20] V. G. da Silva Ruffo et al., "Anomaly and intrusion detection using deep learning for software-defined networks: A survey," *Expert Syst. With Appl.*, 2024, Art. no. 124982, doi: 10.1016/j.eswa.2024.124982.

[21] P. F. de Araujo-Filho, G. Kaddoum, D. R. Campelo, A. G. Santos, D. Macêdo, and C. Zanchettin, "Intrusion detection for cyber–physical systems using generative adversarial networks in fog environment," *IEEE Internet Things J.*, vol. 8, no. 8, pp. 6247–6256, Apr. 2021, doi: 10.1109/JIOT.2020.3024800.

[22] H. Sun, Y. Huang, L. Han, C. Fu, H. Liu, and X. Long, "MTS-DVGAN: Anomaly detection in cyber-physical systems using a dual variational generative adversarial network," *Comput. Secur.*, vol. 139, 2024, Art. no. 103570, doi: 10.1016/j.cose.2023.103570.

[23] O. R. Group, "Datasets used in publications," Accessed: Jul. 10, 2024. [Online]. Available: https://www.uel.br/grupos/orion/datasets.html

[24] UNB, "Ddos evaluation dataset (cic-ddos 2019)," Accessed: Jul. 10, 2024. [Online]. Available: https://www.unb.ca/cic/datasets/ddos-2019.html

[25] UNB, "Intrusion detection evaluation dataset (cic-ids 2017)," Accessed: Jul. 10, 2024. [Online]. Available: https://www.unb.ca/cic/datasets/ids-2017.html

[26] N. Moustafa, "A new distributed architecture for evaluating AI-based security systems at the edge: Network TON_IoT datasets," *Sustain. Cities Soc.*, vol. 72, 2021, Art. no. 102994, doi: 10.1016/j.scs.2021.102994.

[27] J. Donahue, P. Krähenbühl, and T. Darrell, "Adversarial feature learning," in *Proc. Int. Conf. Learn. Representations*, 2017. [Online]. Available: https://openreview.net/forum?id=BJtNZAFgg

[28] I. Goodfellow et al., "Generative adversarial nets," *Adv. Neural Inf. Process. Syst.*, vol. 27, 2014.

[29] C. M. Bishop and H. Bishop, *Deep Learning: Foundations and Concepts*, Springer Nature, 2023.

[30] A. Dunmore, J. Jang-Jaccard, F. Sabrina, and J. Kwak, "A comprehensive survey of generative adversarial networks (GANs) in cybersecurity intrusion detection," *IEEE Access*, vol. 11, pp. 76071–76094, 2023, doi: 10.1109/ACCESS.2023.3296707.

[31] Z. Cai, Z. Xiong, H. Xu, P. Wang, W. Li, and Y. Pan, "Generative adversarial networks: A survey toward private and secure applications," *ACM Comput. Surv.*, vol. 54, no. 6, pp. 1–38, 2021, doi: 10.1145/3459992.

[32] "A review on generative adversarial networks for image generation," *Comput. Graph.*, vol. 114, pp. 13–25, 2023, doi: 10.1016/j.cag.2023.05.010.

[33] H. Lin, Y. Liu, S. Li, and X. Qu, "How generative adversarial networks promote the development of intelligent transportation systems: A survey," *IEEE/CAA J. Automatica Sinica*, vol. 10, no. 9, pp. 1781–1796, Sep. 2023, doi: 10.1109/JAS.2023.123744.

[34] M. Krichen, "Generative adversarial networks," in *Proc. 2023 14th Int. Conf. Comput. Commun. Netw. Technol.*, 2023, pp. 1–7, doi: 10.1109/ICCCNT56998.2023.10306417.

[35] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein GANs," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 5769–5779.

[36] M. L. Proença, B. B. Zarpelão, and L. S. Mendes, "Anomaly detection for network servers using digital signature of network segment," in *Proc. Adv. Ind. Conf. Telecommun./Serv. Assurance With Partial Intermittent Resour. Conf./E-Learn. Telecommun. Workshop*, 2005, pp. 290–295, doi: 10.1109/AICT.2005.26.

[37] H. Liao et al., "A survey of deep learning technologies for intrusion detection in Internet of Things," *IEEE Access*, vol. 12, pp. 4745–4761, 2024, doi: 10.1109/ACCESS.2023.3349287.

[38] B. Yuan et al., "Resource investment for DDoS attack resistant SDN: A practical assessment," *Sci. China Inf. Sci.*, vol. 66, no. 7, 2023, Art. no. 172103, doi: 10.1007/s11432-022-3593-7.

[39] A. M. Zacaron, D. M. B. Lent, V. G. da Silva Ruffo, L. F. Carvalho, and M. L. Proença Jr, "Generative adversarial network models for anomaly detection in software-defined networks," *J. Netw. Syst. Manage.*, vol. 32, no. 4, 2024, Art. no. 93, doi: 10.1007/s10922-024-09867-z.

[40] H. Choi, M. Kim, G. Lee, and W. Kim, "Unsupervised learning approach for network intrusion detection system using autoencoders," *J. Supercomputing*, vol. 75, pp. 5597–5621, 2019, doi: 10.1007/s11227-019-02805-w.

[41] J. Shu, L. Zhou, W. Zhang, X. Du, and M. Guizani, "Collaborative intrusion detection for VANETs: A deep learning-based distributed SDN approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 4519–4530, Jul. 2021, doi: 10.1109/TITS.2020.3027390.

[42] A. Qu, Q. Shen, and G. Ahmadi, "Towards intrusion detection in fog environments using generative adversarial network and long short-term memory network," *Comput. Secur.*, vol. 145, 2024, Art. no. 104004, doi: 10.1016/j.cose.2024.104004.

[43] W. Cheng, Y. Li, and T. Ma, "S-kDGAN: Series-knowledge distillation with GANs for anomaly detection of sensor time-series data in smart IoT," *IEEE Sensors J.*, vol. 24, no. 15, pp. 24344–24354, Aug. 2024, doi: 10.1109/JSEN.2024.3415390.

[44] Y. Wu, L. Nie, S. Wang, Z. Ning, and S. Li, "Intelligent intrusion detection for Internet of Things security: A deep convolutional generative adversarial network-enabled approach," *IEEE Internet Things J.*, vol. 10, no. 4, pp. 3094–3106, Feb. 2023, doi: 10.1109/JIOT.2021.3112159.

[45] N. Abdalgawad, A. Sajun, Y. Kaddoura, I. A. Zualkernan, and F. Aloul, "Generative deep learning to detect cyberattacks for the IoT-23 dataset," *IEEE Access*, vol. 10, pp. 6430–6441, 2022, doi: 10.1109/ACCESS.2021.3140015.

[46] N. Anusha, B. Tapas Bapu, A. Vijayaraj, C. Ramesh Kumar, and R. P, "Cyber intrusion detection using dual interactive Wasserstein generative adversarial network with war strategy optimization in wireless sensor networks," *Multimedia Tools Appl.*, vol. 83, pp. 1–31, 2024, doi: 10.1007/s11042-024-19754-z.

[47] S. Golazad, A. Mohammadi, A. Rashidi, and M. Ilbeigi, "From raw to refined: Data preprocessing for construction machine learning (ML), deep learning (DL), and reinforcement learning (RL) models," *Autom. Construction*, vol. 168, 2024, Art. no. 105844, doi: 10.1016/j.autcon.2024.105844.

[48] R. Mall, K. Abhishek, S. Manimurugan, A. Shankar, and A. Kumar, "Stacking ensemble approach for DDoS attack detection in software-defined cyber–physical systems," *Comput. Elect. Eng.*, vol. 107, 2023, Art. no. 108635, doi: 10.1016/j.compeleceng.2023.108635.

[49] M. R. Kadri et al., "Survey and classification of Dos and DDos attack detection and validation approaches for IoT environments," *Internet Things*, vol. 25, 2023, Art. no. 101021, doi: 10.1016/j.iot.2023.101021.

[50] C. Birkinshaw, E. Rouka, and V. G. Vassilakis, "Implementing an intrusion detection and prevention system using software-defined networking: Defending against port-scanning and denial-of-service attacks," *J. Netw. Comput. Appl.*, vol. 136, pp. 71–85, 2019, doi: 10.1016/j.jnca.2019.03.005.

[51] C. E. Shannon, "A mathematical theory of communication," *ACM SIGMOBILE Mobile Comput. Commun. Rev.*, vol. 5, no. 1, pp. 3–55, 2001.

[52] B. Sharma, L. Sharma, C. Lal, and S. Roy, "Anomaly based network intrusion detection for IoT attacks using deep learning technique," *Comput. Elect. Eng.*, vol. 107, 2023, Art. no. 108626, doi: 10.1016/j.compeleceng.2023.108626.

[53] A. Radford, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015, *arXiv:1511.06434*, doi: 10.48550/arXiv.1511.06434.

[54] T. Schlegl, P. Seeböck, S. M. Waldstein, G. Langs, and U. Schmidt-Erfurth, "f-AnoGAN: Fast unsupervised anomaly detection with generative adversarial networks," *Med. Image Anal.*, vol. 54, pp. 30–44, 2019, doi: 10.1016/j.media.2019.01.010.

[55] B. G. Amidan, T. A. Ferryman, and S. K. Cooley, "Data outlier detection using the chebyshev theorem," in *Proc. 2005 IEEE Aerosp. Conf.*, 2005, pp. 3814–3819, doi: 10.1109/AERO.2005.1559688.

[56] W. Yao, H. Shi, and H. Zhao, "Scalable anomaly-based intrusion detection for secure Internet of Things using generative adversarial networks in fog environment," *J. Netw. Comput. Appl.*, vol. 214, 2023, Art. no. 103622, doi: 10.1016/j.jnca.2023.103622.

[57] H. Gao, B. Qiu, R. J. D. Barroso, W. Hussain, Y. Xu, and X. Wang, "TSMAE: A novel anomaly detection approach for Internet of Things time series data using memory-augmented autoencoder," *IEEE Trans. Netw. Sci. Eng.*, vol. 10, no. 5, pp. 2978–2990, Sep./Oct. 2023, doi: 10.1109/TNSE.2022.3163144.

[58] L. M. Goyal et al., "Deep learning based network intrusion detection system: A systematic literature review and future scopes," *Int. J. Inf. Secur.*, vol. 23, no. 6, pp. 3433–3463, 2024, doi: 10.1007/s10207-024-00896-y.

[59] Z. Abou El Houda, A. S. Hafid, and L. Khoukhi, "MiTFed: A privacy preserving collaborative network attack mitigation framework based on federated learning using SDN and blockchain," *IEEE Trans. Netw. Sci. Eng.*, vol. 10, no. 4, pp. 1985–2001, Jul./Aug. 2023, doi: 10.1109/TNSE.2023.3237367.

[60] Y. Liu, Y. Gu, X. Shen, Q. Liao, and Q. Yu, "MSCA: An unsupervised anomaly detection system for network security in backbone network," *IEEE Trans. Netw. Sci. Eng.*, vol. 10, no. 1, pp. 223–238, Feb. 2023, doi: 10.1109/TNSE.2022.3206353.

[61] A. Komadina, M. Martinić, S. Groš, and Ž. Mihajlović, "Comparing threshold selection methods for network anomaly detection," *IEEE Access*, vol. 12, pp. 124943–124973, 2024, doi: 10.1109/ACCESS.2024.3452168.

Vitor Gabriel da Silva Ruffo received the B.Sc. degree in computer science in 2023 (summa cum laude) from the State University of Londrina (UEL), Londrina, Brazil, where he is currently working toward the M.Sc. degree in computer science and the Ph.D. degree in electrical engineering. Since 2021, he has been a member of the ORION research group with UEL's Computer Science Department. His research interests include software-defined networking, network anomaly detection, and deep learning.

Luiz Fernando Carvalho received the master's degree in computer science from the State University of Londrina, Londrina, Brazil, in 2014, and the Ph.D. degree in electrical engineering and telecommunications from the State University of Campinas, Campinas, Brazil, in 2018. He has experience in computer science with an emphasis on computer networks. He is part of the ORION research group and is currently a Professor with the Federal Technology University of Paraná, Curitiba, Brazil. His main research interests include management and security of computer networks, and software-defned networks.

Jaime Lloret (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees in physics, and the second B.Sc. and M.Sc. degrees in electronic engineering, and the Ph.D. degree in telecommunication engineering (Dr. Ing.) in 1997, 2006, and 2003, respectively. He is currently a Full Professor with the Polytechnic University of Valencia, Valencia, Spain. Since 2017, he has been the Chair of the Integrated Management Coastal Research Institute. He is the Head of the "Active and collaborative techniques and use of technologic resources in the education (EITACURTE)" Innovation Group. He has been Internet Technical Committee Chair (IEEE Communications Society and Internet society) from 2013 to 2015. He has authored 14 books and has more than 650 research papers published in national and international conferences, international journals (more than 375 with Clarivate Analytics JCR). He has been the Co-Editor of 54 conference proceedings and the Guest Editor of several international books and journals. He is the Editor-in-Chief of the *Ad Hoc and Sensor Wireless Networks* (with with Clarivate Analytics JCR) and the international journal *Networks Protocols and Algorithms*. He has led many local, regional, national and European projects. He was the Chair of the Working Group of the Standard IEEE 1907.1 from 2013 to 2018. Since 2016, he has been the Spanish Researcher with highest h-index in the Telecommunications journal list according to Clarivate Analytics Ranking. He is included in the world's top 2% scientists according to the Stanford University List since 2020. He has been the General Chair (or co-chair) of 75 International workshops and conferences. He is a senior member of ACM and a fellow of IARIA.

Mario Lemes Proença Jr. received the M.Sc. degree in computer science from the Informatics Institute, Federal University of Rio Grande do Sul, Alegre, Brazil, in 1998, and the Ph.D. degree in electrical engineering and telecommunications from the State University of Campinas, Campinas, Brazil, in 2005. He is currently an Associate Professor and Leader of the research group that studies computer networks with the Computer Science Department, State University of Londrina (UEL), Londrina, Brazil. He is a master's Supervisor in computer science with Computer Science Department, UEL, where he is also a Ph.D. Supervisor with the Department of Electrical Engineering. He has authored or coauthored more than 120 papers in refereed international journals and conferences, books chapters, and one software register patent. His research interests include computer networks, network operations, management and security, and IT governance. He has supervised more than 150 Ph.D., Master of Science students, graduate and undergraduate students in computer science.