Computer Communications 192 (2022) 299–310

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

FEDGAN-IDS: Privacy-preserving IDS using GAN and Federated Learning

Aliya Tabassum [a], Aiman Erbad [b,*], Wadha Lebda [a], Amr Mohamed [a], Mohsen Guizani [a]

[a] Computer Science and Engineering Department, College of Engineering, Qatar University, Doha, Qatar
[b] Division of ICT, College of Science and Engineering, Hamad Bin Khalifa University, Doha, Qatar

ARTICLE INFO

Keywords:
Deep Learning (DL)
Federated Learning (FL)
Generative Adversarial Network (GAN)
Internet of Things (IoT)
Intrusion Detection System (IDS)

ABSTRACT

Federated Learning (FL) is a promising distributed training model that aims to minimize the data sharing to enhance privacy and performance. FL requires sufficient and diverse training data to build efficient models. Lack of data balance as seen in rare classes affects the model accuracy. Generative Adversarial Networks (GAN) are remarkable in data augmentation to balance the available training data. In this article, we propose a novel Federated Deep Learning (DL) Intrusion Detection System (IDS) using GAN, named FEDGAN-IDS, to detect cyber threats in smart Internet of Things (IoT) systems; smarthomes, smart e-healthcare systems and smart cities. We distribute the GAN network over IoT devices to act as a classifier and train using augmented local data. We compare the convergence and accuracy of our model with other federated intrusion detection models. Extensive experiments with multiple datasets demonstrates the effectiveness of the proposed FEDGAN-IDS. The model performs better and converges earlier than the state-of-the-art standalone IDS.

1. Introduction

The convergence of advanced networking, breakthrough distributed systems technologies, and smart services has rapidly expanded the threat landscape for IoT devices. Researchers have been looking into lightweight and adaptive technologies to solve the problems of cybersecurity in dynamic smart IoT systems, as these domains are increasingly targeted by cyber-criminals, especially smart homes, smart e-healthcare systems and smart cities [1]. Smart e-health devices, also referred as IoT medical devices or Internet of Medical Things (IoMT), record and monitor human vital signs for diagnosis purposes. These tiny devices are empowered with wireless connectivity to treat the patients remotely [2]. IDS is the default and widely used defense mechanisms in IoT devices and IoMTs. IDS demand robust and improved techniques to survive against cutting-edge malicious activities. Unfortunately, the existing IDS developed for IoT devices are based on a strong assumption that the traffic samples available are sufficient, labeled and useful for model training [3]. Besides, a notable amount of research and experimental work substantiates that continuous network monitoring by IDS protects the smart infrastructure the most. Nevertheless, the zero-day attacks, groundbreaking attack techniques and eccentric hackers make any IDS outdated in the face of the novel attacks. To defend against such disruptive malicious activities, the IDS needs continuous improvement at the pace of the variability in the traffic patterns.

Artificial Intelligence (AI) solutions like DL and Machine Learning (ML) have gained enormous attention in the development of anomaly and intrusion detection techniques [4–7]. The research shows that the accuracy of intrusion detection models is proportional to the amount of training data, especially for ML or DL trained models [8]. However, each IoT device has a limited amount of data leading to weak separate models. It is not recommended to collect the local data of IoT devices, particularly for e-health devices as the data is highly-sensitive containing information about the health conditions and other private patient information [9]. The traffic patterns are different on each smart device and can be used to train the IDS. If data of all devices is utilized for training, then the model performance can be improved. However, centralizing the data for training is not feasible due to resource constraints, security and privacy concerns. Because of these two significant obstacles, we look forward for the two advanced technologies with complementary benefits: FL and GANs.

FL models are trained in a distributed fashion locally within devices and only the model updates are shared with the central server [10]. However, it is challenging to build a good federated model with the limited amount of data on each device. We aim to augment the existing data with similar samples to increase the number of samples especially from rare classes, fixing data imbalance issues. A recent successful discovery in ML is the various applications of GANs, particularly in generating synthetic data to increase the number of data samples and to fix under-sampling problems [11]. FL has been applied on discriminative models successfully but there are various on-going investigations on the application of generative networks in FL. Although some of the efforts have been made in modeling GANs in federated settings [12,13], many efforts are needed to model it for a real scenario. We model a GAN network in a federated setting, where local Generators and

* Corresponding author.
E-mail address: aerbad@hbku.edu.qa (A. Erbad).

https://doi.org/10.1016/j.comcom.2022.06.015
Received 29 September 2021; Received in revised form 19 March 2022; Accepted 12 June 2022
Available online 18 June 2022

Discriminators are synchronized periodically by a central Generator and Discriminator with continuous improvement by exchanging gradients and model upgrades. The main contributions of this work are summarized as follows.

(1) Designing a novel distributed GAN-based intrusion detection model for smart IoT devices. The GAN generated synthetic data augments the data on an IoT device to train the IDS model individually. The GAN network solves the problem of limited, missing and imbalanced data on the IoT devices.
(2) Proposing a privacy-preserving FL framework, allowing multiple smart IoT devices to contribute to building a global intrusion detection model. Each device trains its single model on its own data and synthetic local data generated by the local GAN and transfers the model parameters to the global model. The local data and generated data of each device are not shared with other IoT devices. This task ensures data and resources privacy by performing the data preprocessing and model building at each device's own premise. The global model performs parameters aggregation and distributes the updated model to IoT devices.
(3) Developing a comprehensive binary and multiclass classification intrusion detection model after multiple rounds of communication with available IoT devices in the network.
(4) Performing extensive evaluation of the accuracy, performance and convergence of the distributed intrusion detection model with three standard datasets; NSL-KDD and KDD-CUPP99 (KDD99), UNSW-NB15.

The paper is structured as follows: In Section 2, we provide background on GAN and existing studies on IDS. Section 3 presents the main scenario and the framework of the model proposed, while its implementation details are provided in Section 4. Then, a series of evaluation results are shown and discussed in Section 5. Finally, we conclude our work with future directions in Section 6.

2. Background and related work

Before we introduce the proposed framework, we discuss the background and related work available in the literature.

2.1. Generative adversarial network

GAN networks [14] are computationally heavy, as the architecture consists of two deep neural networks called the Generator (G) and the Discriminator (D). In traditional GAN network, G and D are tightly coupled to reach a target learning rate. The Generator trains itself to produce artificial data, while the Discriminator trains to differentiate the original and generated data. The two networks run in parallel to improve their performance gradually. After n iterations, the Generator learns to output data close to the original data and the Discriminator learns to identify the source of the data. This is like a min–max player game framework which can be represented by the function below.

$$\min_{G}\max_{D} = [E_{x\sim P_{data}}[log D_i(x)] + E_{z\sim P_z}[log(1 - D_i(G_i(z)))]] \quad (1)$$

$G_i(z)$ is the synthetic data that Generator produces and the Discriminator output is $D_i[0,1]$, which shows the probability that the data is real or duplicate. The objective of the Generator is to minimize the probability of the Discriminator in identifying the source of data, i.e., $(1 - D_i(G_i(z)))$. While the Discriminator tries to maximize the probability of source identification, i.e., $(1 - D_i(G_i(z)))$ and $D_i(x)$. These networks are powerful in data augmentation, data simulation, anomaly detection, image generation, text-to-image translation [15]. The working of GAN is shown in the Fig. 1.

Mostly, the GAN has always been used in its typical network composed of two neural networks. The input of the Generator is a noise signal, which is a random vector of size k that follows a normal distribution and outputs data similar to the training data. The input to the Discriminator is fed from either the Generator or training sample to differentiate it. Some of the recent research on this [16,17] have modified the GAN architecture into a distributed framework to improve its convergence rate.

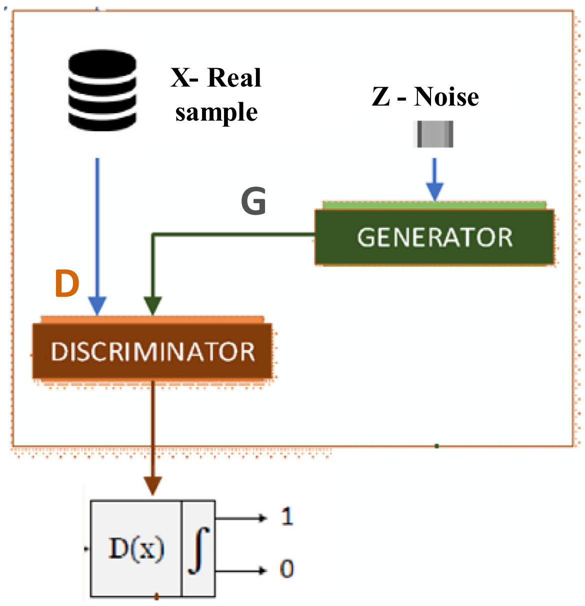

Fig. 1. GAN network.

2.2. Parameter server and federated learning

Some advancements like parameter server approach considered training the non-GAN neural networks on a shared dataset [18]. The parameter server framework, introduced by Google [19] utilizes one or more central servers managing multiple workers and their updated states for parallel processing. The worker networks compute on their local share of data and communicate their weights to a central server. The particularity of DL systems in iterative learning (followed by back propagation) on huge amounts of data mandates parallel processing or distributed training.

A typical GAN architecture consists of two tightly coupled DNNs, a Generator and a Discriminator, nevertheless it can be structured in a distributed fashion adopting a similar method applicable to regular DNNs. FL [20,21] trains DL models on a set of active clients. FL is similar to parameter server framework with a little distinction that clients undergo many local iterations (local training) between each global update (interaction with the central node). Also, some of the workers may become inactive after some rounds. At the beginning of each round, the active clients synchronize the local model updates with the central server. Parameter server framework involves shared data storage. At the beginning of each epoch, the worker machines fetch data from the storage for training. While in FL, the data resides on each worker and the central server does not keep track of any individual worker data and aggregates only the updates sent from the workers to ensure privacy. Because of number of epochs and updates in FL, at each round only one subset of the devices are selected.

In the case of smart home e-health systems, IoT devices are first to produce data. Hence, FL suits best as the on-device data is more pertinent and sensitive.

2.3. Existing IDS for IoT systems

In this section, we review some of the recent studies focusing on intrusion detection based on FL, DL and GAN. Research indicates that IDS gained enormous attention and usage in every domain of IoT environments, right from smart homes, healthcare systems to Cyber–Physical Systems (CPS). Numerous IDS systems were developed based on Deep Neural Networks (DNN). For example, Yang et al. [22] modeled a zone partitioning IDS to identify known and unknown malicious activities in CPS through various zones that have been compromised. Likewise, Yang et al. in 2020 [3] designed an IDS based on Convolutional Neural Network (CNN) for SCADA networks. Among DL models, CNN is highly used to build a strong IDS as the convolutional architecture analyzes the features closely to differentiate minor changes [23]. One of the challenges in improving accuracy of DL based IDS models is selection of optimal features, that can be obtained by appropriate preprocessing techniques. In an attempt by Yazan Otoum et al. [24] proposed an IDS using hybrid pre-processing techniques to select optimal features for training an efficient model. The model achieved higher accuracy and is able to identify anomalous traffic patterns.

Most of these models are trained at a centralized entity which is not only computationally (time and processing) expensive but also threatens the privacy of the data [25]. Though central models trained using ML or DL techniques gives higher accuracy, it overburdens the classifier with huge traffic from all over the IoT devices in the network. Besides, there is a request–response delay in between the model and the IoT devices. Therefore, there was a shift towards distributed, decentralized and similar approaches for IDS. Gajewski et al. proposed a distributed IDS system for smart homes [26]. The data of all IoT devices is collected on a Home gateway (HG) of the Internet Service Provider (ISP) and the IDS is built in 2 levels in a distributed fashion. Level 1 is the IDS at the HG which analyzes the traffic of all IoT devices and gives alerts. While level 2 is the Network-IDS provided by the ISP which re-analyzes any malicious alerts by the IDS at HGs.

Nevertheless, decentralized models proposed in literature, transfers the original data to other sources, which is vulnerable to be intercepted or leaked. Recently, FL was adopted for IDS to address centralization and isolated data issues [27,28]. FL is a distributed ML approach involving Edge computing. FL is similar to distributed learning and trains the model locally so it does not share the data with other devices ensuring data privacy. Nguyen et al. [29] proposed a self-learning on-device IDS using FL for IoT devices. They modeled the IDS as an anomaly detection module that alerts any communication deviations from the regular pattern in the IoT devices. Likewise, Yulin et al. [30] designed IOTDefender, a IDS framework for 5G IoT devices using FL. The model is built in collaboration with all available IoT devices by transfer of parameters without privacy leakage. The model achieved 91% average accuracy with lesser false-positive rate than other unified DL models. Another IDS model is built by [31] using Gated Recurrent Units (GRUs), which also employed regular federated scenario by sharing computed weights with the central server.

The FL based IDS proposed in the literature outperforms the classic centralized and distributed models, but those models are vulnerable to various attacks [32,33]. Federated approach leaves some backdoors for attackers to manipulate the training process or to compromise the model built [34]. In the race of obtaining better models, FL fails to inference and similar attacks [35,36]. Attempts have been made to defend federated models against poisoning attacks using adversarial networks [12,37,38]. If the training data is mixed up with some other synthetic data, that computes different parameters and it becomes difficult to launch membership inversion and inference attack to gain the actual data set or a sample from the training set. In addition, the order of transfer of parameters and aggregation techniques changes the model accuracy. Besides, models trained using FL give poor performance and higher false alarm rates, if there is limited training data. Considering these arguments, we investigate IDS models build using GANs.

Seo et al. [39] proposed a GAN-based driver safety system to reduce the false alarm rates in Vehicular Networks (VN) by augmenting the training data. In a fully decentralized way, Ferdowsi et al. [40] designed an adversarial network to identify anomalies in IoT devices.

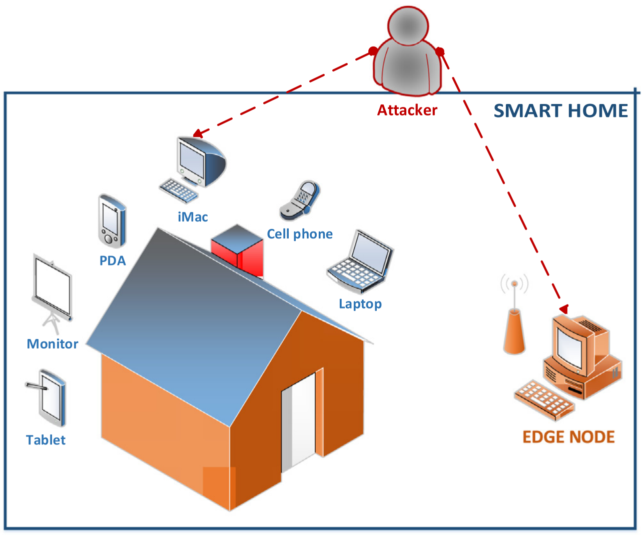

Fig. 2. Smart home IDS scenario.

This model was aimed to conceal the user's local data (IoT private data) by using a single Generator in the network and a Discriminator network on each IoT device. However, the central Generator collects the data distributions of all IoT devices in the network. The central Generator for data analysis is a major bottleneck for scalability and request response. There are many GAN based IDS models and federated IDS models but none of them is designed using GAN in a federated scenario for intrusion detection. We engage GAN in our model to synthesize training data to improve the model's accuracy by augmenting synthetic data. As the data augmentation ensures less false-positive rates and better learning for minor class samples [41]. FL distributes the workload by enabling multi-party on-device learning. This ensures the privacy of data while allowing the data available in each device to be used in the training process. Besides, synthetic data mixed with original data increases the robustness of the federated model, as the synthetic data is generated from a noise vector.

3. Proposed framework

In order to show the application of the proposed framework, we consider a smart IoT system scenario, such as a smart home consisting of various IoT and e-healthcare devices, as shown in Fig. 2. Parts of the data from the IoT devices are sent to the Edge and o the external network, cloud servers, for further processing and analysis. Any health data could be sent via the cloud to the medical practitioner or other participants.

The components of the system that we focus are as follows:

(1) *IoT devices or Clients*: The source of data generation which needs to be protected against privacy and security threats.
(2) *Edge Node*: The Edge node is a mediator between the cloud and the IoT devices. It serves as a central entity for FL and needs protection to secure the model and also data residing on the Edge node.

The system is composed of a set of N IoT devices i.e., clients or worker machines, each consists of a local dataset B^n of size m and the number of features d that are transmitted through it. All the devices in that particular network follow a probability distribution $P_{data}(x)$, where x is Non-IID (Independent and Identically Distributed) i.e., time-series data, health records, temperature monitoring, or financial data. A set of data is said to be IID if the probability distribution of all random variables is same. In this scenario, the probability distribution of the

Table 1
Notations.

Notation	Description
D^n	Local Discriminator
G^n	Local Generator
D^c	Central Discriminator
G^c	Central Generator
N $= 1, 2, 3\ldots, n$	All devices/Client in the network
S_t = 1,2,3....,m	Available client/active set
w_t	Generator parameter at epoch t
θ_t	Discriminator parameter at epoch t
K	Synchronization interval
i..... I	Local training iterations
t = 0,1,2, ... E	Global training epochs
B^n	Local dataset on a device
ℓ^n_g	Loss of Generator of client n
ℓ^n_d	Loss of Discriminator of client n
E^n_g	Error feedback of client n on Generator
E^n_d	Error feedback of client n on Discriminator
b^n	batch size local
d^n	Size of dataset (feature set size) on a device
m^n	Number of samples in a dataset on a device
α_d	Learning rate of Discriminator
α_g	Learning rate of Generator
λ	Penalty coefficient

data on each IoT device may be different [42]. The entire dataset of all active agents is denoted as $B = \cup_{n=1}^{N} B^n$, where B^n is the local dataset of each IoT device n. We aim to build an IDS to protect the IoT devices and Edge device from internal i.e., within the network and external attacks. Apart from attacks that target IoT devices, there are some attacks targeted towards the Edge node. If we build a model by collecting all the data at one place, we endanger the privacy of the critical data on those devices. Besides, data centralization increases the communication overhead and the data can also be easily manipulated at one central entity. Thus, the local datasets in each worker should remain in place and should not be sent for training on other devices.

3.1. Problem formulation

We formulate the intrusion detection problem for multivariate time series as follows. In the first phase, the GAN model is trained based on the time-series dataset of each IoT device, X-input, and generates "similar" samples X_G that "look like the dataset". We model the Generator function as G_w: $R^l \rightarrow$ X, R and l are fixed. Likewise, the function representing the Discriminator is D_θ: $X \rightarrow$ [0,1] where $D_\theta(x)$ defines the probability such that x is a normal sample, while θ is the parameter of the Discriminator. According to the parallel theory [43], the model passing through various clients by exchanging the model parameters does not expose the privacy of other clients. The local Generators analyze the local traffic and generates similar traffic pattern, but only the gradients of the local Generators are transferred to the central Generator. So, the output of the local Generators on each client guarantees differential privacy as per the explanation provided in [44]. In every global communication round the parameters need to be exchanged among the clients, which gives the communication complexity for one round as $(|w| + |\theta|)$. The number of parameters passed also depends upon the number of Generators and Discriminators active in a communication round. So, the total communication complexity for all rounds is N.$(|w|+|\theta|)$. One major advantage of the proposed framework is the Edge device which is placed nearby to the IoT devices that ensures better communication efficiency. The notations used in this paper are shown in the Table 1.

3.2. Proposed architecture

We first introduce distributed GAN setup for Intrusion detection and adaption of FL. The complete architecture of our model is shown in Fig. 3. Our framework consists of a central GAN on Edge and a local GAN placed on each of the IoT device. The two central networks (Generator and Discriminators) of the central GAN are responsible to aggregate all the gradients received from the independent Generator and Discriminator networks of the local GAN on each IoT device. The communication between these networks can be visualized in Fig. 3.

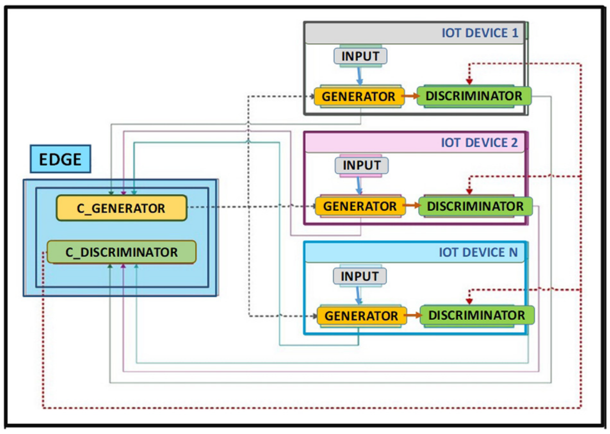

Fig. 3. FEDGAN-IDS model training architecture.

Our approach is inspired by MDGAN (Multi-Discriminator Generative Adversarial Networks) [45] and FL-GAN [44] models, which are proposed for synthetic image generation and privacy-preserving, respectively. MDGAN consists of single Generator at the central level and multiple Discriminators at the worker levels. The data from the single Generator is shared by splitting over the workers. During training the Discriminators are swapped in between workers in a peer-to-peer fashion while in our model they stay fixed and are averaged by the server upon reception from the workers. Besides, we have independent Generators on each client which ensures privacy of the data as per the proof provided by FL-GAN architecture. Though the single Generator at the central level is suitable for image generation, the analysis of input data at one place threatens privacy and slows down the classification process.

FL-GAN has generators and discriminators at the workers as well as the server and uses federated learning to distribute data with updates that are averaged by the server. By design FL-GAN has a Generator and a Discriminator tightly coupled which adapts to a known computation method by parameter server approach on clients and server side.

Both MD-GAN and FL-GAN architectures have been proposed for image processing so the machine learning models used are different from the one we have developed for IDS. In those works, the entire GAN network was modeled to perform an image dataset augmentation task. Whereas in our work we modeled the GAN network to perform two different tasks of (1) sample generation at the IoT nodes similar to them and (2) also perform an innovative anomalous traffic classification task. Though the similar architecture is adopted, it is considered in the context of distribution of IDS over various independent IoT devices. No model was designed to derive benefits from combined architecture of FL and GAN for IDS models. The Generator network in each IoT device is used for generating similar network traffic samples as the traditional way of GAN modeling. However, the Discriminator network identifies anomalous samples from the overall samples that are provided from the Generator network as well as the IoT input traffic. The discriminator is used as the classifier. We have designed both the binary and the multiclass classification model.

3.3. Distributed IDS using GAN

We build a distributed IDS model on IoT devices using GAN. Each IoT device has two networks, namely, Generator and Discriminator. The

Generator is a deep neural network that analyzes the local traffic of the IoT device and generates a similar traffic pattern. The synthetic data from local Generator and original traffic of the IoT device are combined and provided to the Discriminator network for training a classification model. The Discriminator is a CNN that identifies malicious traffic patterns in the samples fed to it. Among DL models, CNNs have gained enormous success in the identification of malicious traffic in time-series data [23,46]. Each IoT device builds its own local model on the Discriminator network. The DNN networks give higher accuracy with a good amount of training data. With that motivation, we have incorporated a Generator network that generates similar traffic on each local worker and that augments the available data to train the Discriminator used in the IDS model. A central Generator G^c and Discriminator D^c is hosted on an Edge server in the smarthome network. Likewise, local Generators G^n and Discriminators D^n are hosted by workers. The central Generator and Discriminator hold the initial parameters w and θ respectively G^c_w, D^c_θ. Each worker starts training its local Generator G^n with the initial parameter w_0 (initially received from the central Generator) on its local dataset B^n. Similarly, the local Discriminators D^n are trained with the initial parameter θ on its local dataset and data generated by the local Generator $B_n + X_{G^n}$.

3.4. Federated learning framework

In conventional FL, a subset of existing devices participate in FL. Each device loads the central model and computes it based on its local data. Then each device sends its updated model to the central model. The server aggregates these local models to construct an improved (updated) model. Likewise, the proposed federated structure aims to collectively build a distributed IDS. There are two central models: a Generator and a Discriminator. Initially, the central models select parameters: weights (w, θ), learning rate (α_g, α_d), batch size b, penalty coefficient (λ) and decay rate. Then, a ping message is sent to select the active devices in the network to send the parameters. Finally, the parameter sets of the Generator and the Discriminator are sent initiating a training epoch. The initial architecture and parameters of the local Generators and Discriminators may be changed after their first global epoch.

There are numerous challenges for the efficient training of our model. The coupling between a Generator and a Discriminator requires organized strategies between the clients (devices) and dictates that the computational load on the devices be rational. Initially, each Generator creates synthetic data by analyzing the probability distribution of the local traffic on the device. The Discriminator takes the input from the local device and the local Generator to train its network for intrusion detection. After completion of local iterations on two deep neural networks, the gradients are transferred to the central networks. The local model parameters aggregation is performed by the central model and then the updated central model is shared with and adopted by the end devices. The central Generator and Discriminator collect the gradients from all local G^n and D^n and aggregate them. The updated gradients are communicated back to the local networks on the devices. The local Generators receive the updated gradients to build a new model in order to improve the quality of samples generated. Similarly, the local Discriminators improve its detection accuracy by the updated gradients from the central Discriminator.

4. FEDGAN-IDS algorithm

It is a federated generative adversarial framework for training an IDS across distributed IoT devices to preserve the privacy of the Non-IID data. In our algorithm, a global communication round trains the local Generators and Discriminators on each IoT device, that are periodically synced via a central Generator and Discriminator on the Edge that aggregates and transmits the new parameters to all local Generators and Discriminators.

A global communication round is composed of 4 phases:

- Phase 1 – Local Generator Training
- Phase 2 – Local Discriminator Training
- Phase 3 – Aggregate parameters at central model
- Phase 4 - Model Parameters Dissemination

The detailed explanation of each of these phases is provided in the below subsections.

4.1. Phase 1 : Local generator training

The training begins from the local Generators. Initially the local Generator receives parameters (w_0 and α_g) from the central Generator to begin training. Each local Generator captures the real data distribution from the IoT device and generates similar traffic patterns. The number of local Generators is equal to the number of available users (IoT devices) in the network. The Generator is improved gradually with its loss at every local iteration. The number of samples is denoted by m and whereas the number of features is denoted by d. The training goes on for I number of local iterations with respect to the sample generation loss ℓ_g. The outcome of this network is a similar set of traffic pattern X_G. The number of samples generated balance the rare classes such that the normal and anomaly patterns are in similar proportions. The working of the local Generator is shown in Algorithm 1. After completion of local iterations, the gradients of Generator network and error feedback E^n_g is sent to the central Generator. Error feedback of Generator reports its amount of error in data generation on each device. The generated synthetic traffic is mixed up with the real data of IoT and fed to the local Discriminator for training.

Algorithm 1 - Local Generator

```
Input:
 1: Receive $w_0$ from the central Generator.
 2: Receive $B^n$ from local device.
 3: Set local iterations (I)
Output: Synthetic data from each Generator
 4: procedure LocalGenerator($B^n$, $w_t$)
 5:     for i ⟵ 1 to I do
 6:         $Z_i$ ⟵GAUSSIAN NOISE (b)
 7:         $X^n_G \longleftarrow G^n(w_t, z) \mid z \in z_i$
 8:     Calculate $w^n_i$
            $$w^n_i \Longleftarrow Adam((\frac{1}{m}\sum_{i=1}^{m} \Delta_w L^i + n), w_i, \alpha_g) \quad (2)$$
 9:     Calculate $E^n_g$
            $$E^n_g \Longleftarrow \frac{\partial B(X^n_G)}{\partial x_i} |x_i \epsilon (x^n_G) \quad (3)$$
10:     Calculate $L^n_g$
11: end procedure
12: Send : $w_i$, $E^n_g$ to the central Generator
```

4.2. Phase 2: Local discriminator training

Step 2 is the process of training Discriminator for I number of local iterations as shown in Algorithm 2. The number of local Discriminators is also equal to the number of users (IoT devices) in the network. Local Discriminator trains on original traffic and generated traffic of the local Generator to enhance the training process with enough number of samples. The data is pre-processed before training the intrusion detection model. Local Discriminator incorporates weights, biases, penalty term and L2 regularization to overcome the problem of overfitting. The outcome of the network is the classification of the sample as attack or normal for a Discriminator network designed for binary classification. Whereas, for multiclass classification network it gives outcome as per the class of attack category. We also record the training accuracy and testing accuracy along with the loss of the Discriminator ℓ_d. After the training process (local iterations) of each client, it computes an error feedback of the Discriminator network E_D. The computation of this function is shown in Algorithm 2. Discriminator error feedback reports its false-positive and true-negative rate in anomaly detection process.

Algorithm 2 - Local Discriminator

```
Input: Receive $\theta_0$ from the central Discriminator. Receive $X^n_G$ from
    local Generator of the device. $B^n$ local dataset from the device. Set
    local iterations (I)
Output: Normal or Anomaly
1: procedure LocalDiscriminator($B^n$, $X^n_B$, $\theta_t$)
2:     for i ⟵ 1 to I do
3:         $D^n = X^n_G + B^n$
4:         Disc Learning ($J_{disc}$, $D^n$)
5:         Update Discriminator by ascending the stochastic gradient
               $\theta^n_i \Longleftarrow Adam(\Delta_\theta \frac{1}{m}\sum_{i=1}^{m} -D(G_\theta(z)), \theta, \alpha_d)$    (4)
6:     Calculate $E^n_d$
               $E^n_d \Longleftarrow \sum T.N + F.P$    (5)
7:     Calculate $L^n_d$
8: end procedure
9: Send : $\theta^n_i$, $E^n_d$ to the central Discriminator
```

4.3. Phase 3: Central model update

The Algorithm 3 demonstrates the training of this phase. The central Generator and Discriminator receive the parameters (w_t, θ_t) and the error feedbacks E_G and E_D from all client models. The parameters received undergo federated averaging and gradients calculation to get the updated model parameters to be sent to all the active clients. The two central networks broadcasts the w_{t+1}, θ_{t+1} to all clients. We perform E global epochs until the satisfactory accuracy is achieved on all Discriminators. The central models tries to minimize the error feedback of Generators and Discriminators at each global epoch.

Algorithm 3 - Central GAN

```
Input: Initialize $\theta_0$ for $D^n$. Initialize $w_0$ for $G^n$. Set Global epochs (E).
    Set the learning rate $\eta$ at each epoch $\alpha(\eta)$ and b($\eta$) synchronization
    interval $K^t$.
Output: Updated parameters
1: for t = 1,2,...,E do
2:   Set of available clients: m
3:     for m $\epsilon$ $s_t$ do
4:         → Local Generator ($w_t$)
5:         → Local Generator ($\theta_t$)
6: /** Each Client trains locally in parallel **/
7: if (t mod K) = 0 then
8:     clients transfer gradients to central server
           $w_{t+1} \overset{\Delta}{=} \sum_{n=1}^{m} P^n W^n_t$    (6)
           $\theta_{t+1} \overset{\Delta}{=} \sum_{n=1}^{m} P^n \theta^n_t$    (7)
9: end if
10: Send the updated parameters to all clients; $w_t + 1$ and $\theta_t + 1$
```

4.4. Phase 4: Model parameters dissemination

The updated model parameters are sent back to all available clients to build their new models. In the testing phase of this framework, only the local Discriminators are present for classifying the traffic on IoT devices and also the central Discriminator on the central Edge. After empirically determined E epochs (global epochs) a final IDS model is obtained.

5. Performance evaluation

In this section, we evaluate the performance of the FEDGAN-IDS model in terms of accuracy, loss, recall, precision, F1-score, AUC score and convergence rate. We also illustrate the results recorded for binary and multiclass classification of the proposed IDS model. Besides, we compare the performance of a FL based IDS (FED-IDS) built using same hyper-parameters without the GAN architecture. The number of epochs for communication rounds (global) is kept constant for accurate comparison. We perform all experiments using three standard datasets: NSL-KDD, KDD99, UNSW-NB15. We perform all the necessary pre-processing steps for training the Discriminator models. The experiments are conducted on a Intel(R) Core (TM) i7 - 10750H CPU @3 GHz predator machine with 4 GB NVIDIA GeForce RTX2060. For each global communication round, all the local models parameters are transmitted to the central Edge server for aggregation and update.

5.1. Datasets description

(1) KDD-CUP99: It is one of the popular and widely used dataset for network IDS and it was made public in 1999. This dataset is improved from DARPA 98 dataset [47]. However, it has redundant records. Even though the dataset has some shortcomings, we decided to use it in testing to ensure the non-biased nature of the proposed model and the efficiency of the model in eliminating the duplicate features. This dataset has 24 attack classes and 41 features. However, the attack classes are highly imbalanced. In a typical binary classification model this dataset achieves higher accuracy since only 2 class categories are divided (Normal and Anomaly). When multiclass classification model is tested, the accuracy of the model is very low due to largely imbalanced data. This is another significant reason for choosing KDD99 to show the performance of our model with an imbalanced dataset. The 10 labels of KDD99 dataset are categorized into 4 standard attack classes, Denial of Service (DoS), Remote to local (R2L), User to root (U2R) and Probe.

(2) NSL-KDD: It is an improved version of KDD99 in which the duplicate features were eliminated. This dataset was made public in 2009 and from then it was used for numerous network IDS evaluation and testing [48]. NSL-KDD is also used in evaluation of IDS for IoT devices using FL [49]. Similar to KDD99, this dataset has 41 features and various attacks which can be categorized into 4 standard attacks: U2R, R2L, probe and DoS. The normal samples constitute 53% of this dataset and the number of different attack samples are highly uneven, making it imbalanced and unsuitable to test for multiclass models.

(3) UNSW-NB15: UNSW-NB15 is one of the comprehensive publicly available datasets which was published in 2015 by Mustafa et al. [50]. From the all collected detailed traffic 10% was included in the dataset. A large amount of IDS models developed have utilized this dataset for training and testing purposes. For instance, Ahmad et al. [51] have used UNSW-NB15 dataset for a supervised ML IDS for IoT devices. The dataset has 9 attack classes, namely, Fuzzers, Backdoors, DoS, Generic, Reconnaissance, Analysis, Exploits, Shellcode, and Worms. The attack classes of UNSW-NB15 are not changed and we have tested it with 10 labels in multiclass classification scenario.

5.2. Data pre-processing

The data is pre-processed before the classification task. In the pre-processing step we perform the following tasks.

(1) Encoding: The categorical values in each of the dataset are transformed into numeric features. In case of UNSW-NB15, there are 41 numeric values and 3 strings. The string values are encoded into numeric to train the model. Likewise, the categorical values in NSL-KDD and KDD99 are also transformed into numeric.
(2) Normalization: We have performed feature scaling to ensure that all features are in uniform magnitude and range by using the euclidean distance between two data points.
(3) Feature extraction: Usually known as feature reduction, which can be done by linear or non-linear transformation. CNN is capable to identify optimal features from the existing data. The non-linear architecture of CNN helps to reduce the dimensions from a high-dimensional space. In addition, the convolutional layers eliminates the redundant features to make the model more simple and effective. We have used 41 features of KDD99 and UNSW-NB15 dataset in all models for experimentation. Whereas, in case of UNSW-NB15 dataset 45 features are used.

5.3. GAN network specifications

The specifications of the GAN-IDS network are as follows:

(1) Initially the training samples are transformed from 1D feature vector into a 2D image form, since GAN networks achieves highest performance with the 2D image data [52]. Each 1D network flow data is converted into one 2D gray-scale image, i.e., aXb two-dimensional feature matrix.
(2) The Generator deep neural network is trained similar the Generator of supervised Auxiliary Classifier GAN (ACGAN) by following same data augmentation procedure [53]. It learns the distribution of the image training set to generate new artificial attack samples in order to balance the dataset. The Generator does not memorize the training data as the nearest neighbors from the training data are not matched with the synthetic data generated. The supervised Generator contributes in generating high quality data. By using the labels, we directly apply cross entropy loss on the Discriminator, which adds extra label in identification of normal and anomalous pattern.
(3) The Discriminator is modeled as a 2D CNN which is trained on the augmented training set to identify normal and anomalous patterns. The input nodes of the first layer of Discriminator network is based up on the output of the Generator network. The Discriminator network is consist of a first 2D convolution layer, 2D pooling and fully connected layer with a 3×3 matrix of the convolutional kernel. We have tested different parameters for tuning the CNN. Finally, we chose these parameters for our network. We incorporated 32 filters for convolutional operation with the size of filter 5. Batch size selected is 32 with "LEAKY RELU" activation function. The 2D pooling layer used max-pooling technique with two strides. To avoid overfitting and over learning of the Discriminator network, we apply L2-regularization and Dropout 0.5 before the output layer. The loss is calculated by the generated output and the actual output based on cross-entropy function. To increase the stabilization ability of the Discriminator network, we use spectral normalization on the weights of the layer.

5.4. FED-IDS binary classification model

In the first case, we model the IDS using FL without GAN. In order to compare FEDGAN-IDS, we model this adapted version of FED-IDS. In this model, we place a central Discriminator network and each independent local Discriminators on the IoT devices. The local discriminators are trained on the local data of the device, without any augmentation. The three datasets NSL-KDD, KDD-CUP99 and UNSW-NB15 are divided into 3 sets: Train, Validation and Test with 70%, 15% and 15%, respectively. From the training sets the normal attack samples are equally distributed on all available IoT devices. Whereas the attack samples are randomly distributed on all IoT devices. However, there is no augmentation of the data in this case, since there is no Generator network. After each epoch of training, the gradients of local Discriminators are transferred to the central Discriminator. The central network aggregates and sends back the updated gradients. After each global round, more data samples are provided which ensures better training and improved accuracy. Likewise the process continues for 20 global epochs and after which we record the results on a Discriminator which performs its local iterations on its local dataset. The accuracy and loss of this model is shown in Figs. 4 and 5, respectively. The accuracy was recorded on an IoT device for three datasets: KDD99, NSL-KDD and UNSW-NB15. We observe that the accuracy is gradually increasing with the number of epochs.

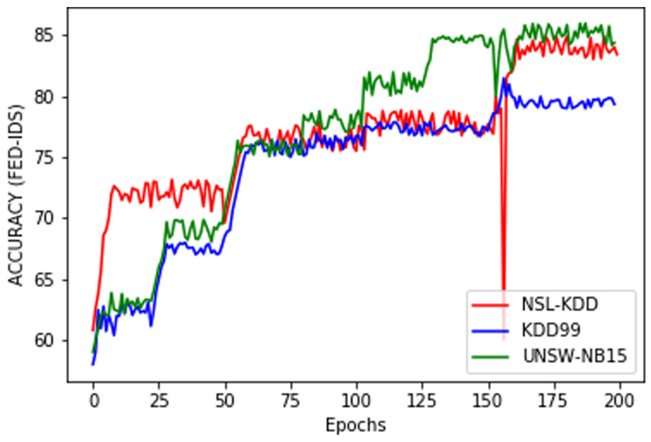

Fig. 4. FED-IDS Binary classification accuracy. The accuracy of a Discriminator model after 20 global updates using three datasets.

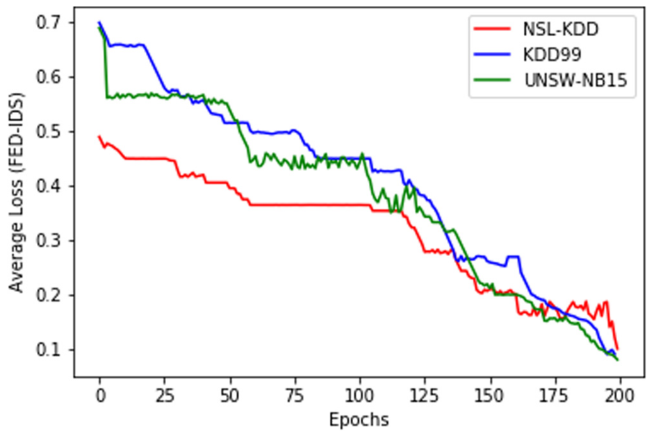

Fig. 5. FED-IDS Binary classification loss. The loss of a Discriminator model after 20 global updates using three datasets.

5.5. FEDGAN-IDS Binary classification model

In the second case, we model the IDS using FL and GAN. Each IoT device has two DNNs, Generator and Discriminator. The training samples division is similar to the first case for three datasets. From the training sets the normal attack samples are equally distributed on all

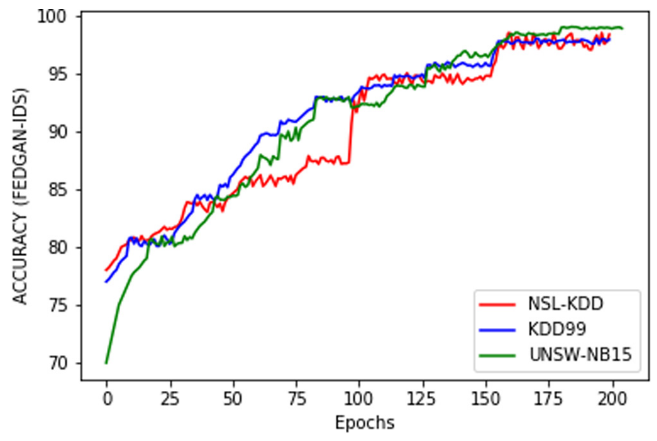

Fig. 6. FEDGAN-IDS Binary classification accuracy. The accuracy of a Discriminator model after 20 global updates using three datasets.

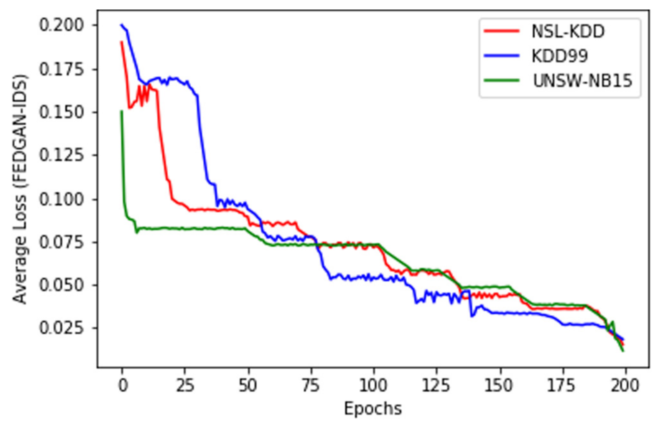

Fig. 7. FEDGAN-IDS Binary classification loss. The loss of a Discriminator model after 20 global updates using three datasets.

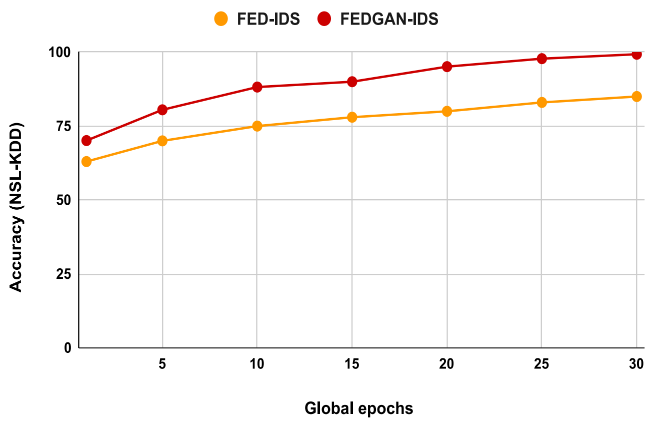

Fig. 8. Model convergence of Fed-IDS and FedGAN-IDS using NSL-KDD dataset.

available IoT devices. Whereas the attack samples are randomly distributed. In Binary classification (normal and anomaly), the Generator network augments the data such that whatever percentage of normal samples an IoT device gets, the same proportion of attack samples are generated to have a balanced local data for training. For instance IoT device 1 gets 15% of normal samples from the whole dataset then the attack samples are augmented such that it constitutes 15%. The Discriminator networks trains on the augmented data (original and generated data). Once the Generator and Discriminator models are built, the parameters of each model are transferred to the central nodes for aggregation from all IoT devices. After 20 global epochs, we have recorded the accuracy and loss of the Discriminator model. The accuracy of the model is shown in Fig. 6. The loss of this model is shown in Fig. 7. As the data is Non-IID, we observe some deviations in the results. The local iterations of each local model is considered 200, whereas the global epochs are chosen as per the performance of the models. After every global communication round of the FL phase, the updated gradients improve the network.

The accuracy of both models (FED-IDS and FEDGAN-IDS) increases gradually with the local iterations on the IoT device. However, the accuracy of the model without GAN network lies between 75% to 85% for three of the datasets. While adding the GAN network helps to improve the accuracy at an earlier stage in the federated scenario. In FL with a GAN network, the model performance is enhanced in terms of accuracy and loss. The accuracy recorded is greater than 97% for all the datasets tested. Moreover, in FED-IDS case, the loss of the model moderately decreases and it reaches it lowest value 0.1. On the contrary, the loss is almost negligible (0.02–0.01) in FEDGAN-IDS case for the three datasets.

After which, we show the two models convergence in Fig. 8 for NSL-KDD dataset. After every global epoch, both models improve in terms of accuracy. At the epoch 10, FED-IDS model achieves the accuracy of approximately 74% whereas FEDGAN-IDS model achieves more than 85%. The convergence state is reached 5–10 epochs faster in GAN case than the model without GAN. FED-IDS reaches its maximum value 78% after 25 epochs and no significant enhancement is seen after that. With GAN network, the model reaches more than 95% of accuracy after 15 global epochs and the detection accuracy is 10%–12% higher in FEDGAN-IDS case. We have similar results with the other two datasets, which can be observed from Fig. 12. We deduce that adding GAN network to FED-IDS is promising for better performance of IDS model with FL. The augmented data by GAN ensures enough training samples which improves the accuracy of the overall model.

We also show the performance evaluation of our FEDGAN-IDS model in terms of training and testing accuracy of local Discriminators on its local dataset (own data + synthetic data generated by local GAN). The training accuracy of the three Discriminator models is recorded at the global epoch 3 i.e., after 3 global updates using NSL-KDD dataset. The Discriminator models are trained successfully and each of the local models give nearly equal accuracy. We record the training and testing accuracy of all local models during the local iterations (200) and also during each global epoch. The test accuracy of 3 Discriminators on the 15% of the local dataset is shown in Fig. 9. The Test accuracy of 3 Discriminators on different dataset (other than NSL-KDD samples) is shown in Fig. 10. At the beginning of the 3rd global epoch, the test accuracy on the local Discriminators is recorded from 0 to 200 local iterations. The test accuracy starts with a very low value and gradually increases to a satisfactory accuracy level with the local dataset. Whereas with different datasets there are many deviations in the results. The reason for testing the Discriminator models performance with different datasets is to confirm that our model is not over-fitting. We incorporated the validation dataset to ensure the non-biased nature of the model.

Moreover, the accuracy of local Discriminator models is recorded while testing with the NSL-KDD dataset for 9 global epochs. It is illustrated in Fig. 11 for FEDGAN-IDS model. The detection accuracy of three Discriminator models ($Discriminator(i)$) is recorded for different global epochs ($G\_iter(i)$). FEDGAN-IDS models are trained on the augmented data of the IoT device and parameters are sent to the central model. After parameter aggregation, the updates are communicated back to all DNN models which are re-trained on the available local data of the IoT devices. After each global epoch, the test accuracy of the individual model is recorded to understand the performance of the model. We have also recorded the individual Discriminators performance using KDD99 and UNSW-NB15 in Fig. 12. With these

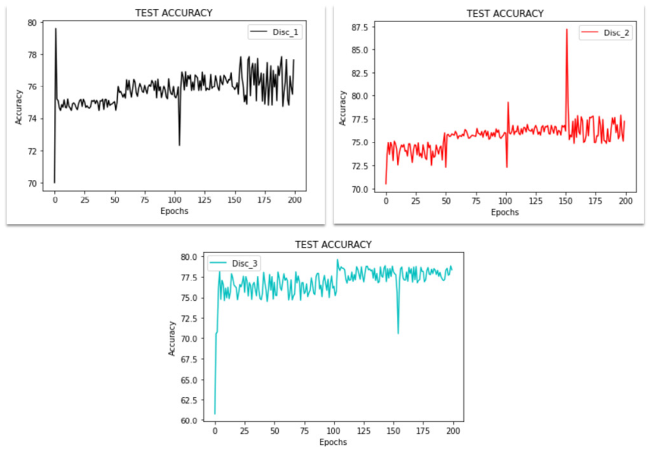

Fig. 9. FEDGAN-IDS test accuracy on local discriminators at global epoch 3 with local test data.

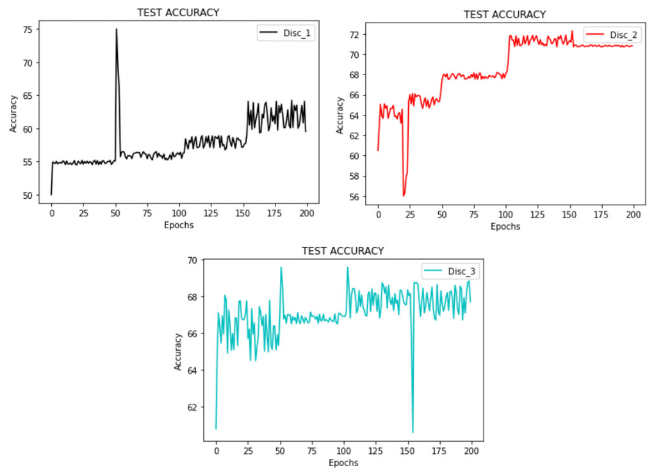

Fig. 10. FEDGAN-IDS test accuracy on local discriminators at global epoch 3 with different dataset.

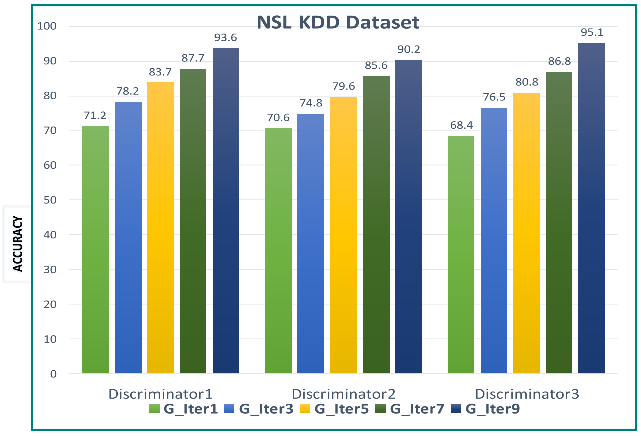

Fig. 11. FEDGAN-IDS Binary classification test accuracy at each global epoch on different discriminators using NSL-KDD dataset.

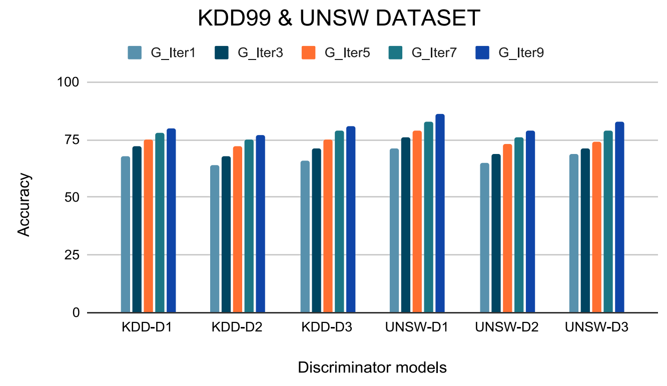

Fig. 12. FEDGAN-IDS Binary classification test accuracy at each global epoch on different discriminators using KDD99 and UNSW-NB15 dataset.

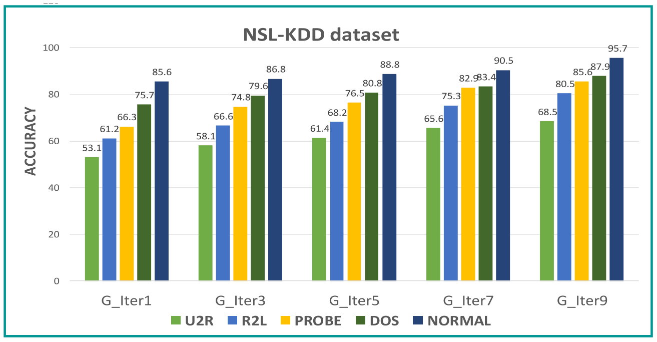

Fig. 13. FEDGAN-IDS multiclass classification: Accuracy of different classes for first 9 global epochs using NSL-KDD dataset.

two datasets also, we have recorded for three Discriminator models, $KDD - D(i)$ and $UNSW - D(i)$ for 9 global epochs, $G\_Iter(i)$.

From the results, we observe that test accuracy improves with every update on all Discriminator models using three datasets. We are concerned with the attack detection accuracy of the IDS model, hence we have recorded the performance of the Discriminator models. The results signifies continuous improvement of the Discriminator models at each epoch with better accuracy than the FED-IDS model. The summary of results for Binary Classification FEDGAN-IDS model using three datasets are shown in the Table 2. The comparison of FED-IDS (adapted) and FEDGAN-IDS (proposed) in terms of all metrics is also shown.

5.6. FEDGAN-IDS multiclass classification

In the third case, we modeled the Discriminator as a multiclass classifier network to identify different categories of attacks. We have followed similar division of the training dataset. The normal samples are divided equally on all available IoT devices. Whereas, the attack samples are randomly distributed. In this case, the attack samples are specifically categorized with their class labels such as (DoS, Probe, U2R and R2L in the case of NSL-KDD and KDD-CUP99). Each IoT device may get more than one type of attack samples. We balance the dataset considering the attack samples and also the normal samples. The number of attack samples augmented by the Generator are proportional to the percentage of number of normal data samples that an IoT gets, such that the overall data is balanced on that IoT device. For example, IoT device 1 gets 15% samples of DOS and 15% of Probe compared to the normal samples it has. Then the two attack samples are augmented such that entire dataset of IoT device 1 is balanced. The similar division pattern is followed for UNSW-NB15 dataset. The selection of hyper-parameters played a principal role in training a multiclass FEDGAN-IDS model. The decay rate is selected as $1.0X10^{-4}$ and the penalty coefficient is chosen 0.05. After the models are trained the parameters of Discriminator and Generator from all IoT devices are transferred to the central Generator and Discriminator for aggregation.

Table 2
Binary classification comparison of FED-IDS and FEDGAN-IDS with three datasets.

Metrics	NSL-KDD		KDD99		UNSW-NB15	
	FED-IDS	FEDGAN-IDS	FED-IDS	FEDGAN-IDS	FED-IDS	FEDGAN-IDS
Accuracy	85.3	99.29	83.8	99.1	83.5	99.4
Precision	84.1	99.3	80.5	97.8	83.8	99.56
Recall	79.2	98.9	78.3	96.3	85.9	99.3
F1-Score	82.8	99	82.2	98.5	84	98.9
AUC-Score	78.4	99.01	79	98	84.6	98.78

Table 3
Multiclass classification of FEDGAN-IDS using NSL-KDD and KDD99.

Metrics	U2R		R2L		PROBE		DoS		Normal	
	NSL-KDD	KDD99	NSL-KDD	KDD99	NSL-KDD	KDD99	NSL-KDD	KDD99	NSL-KDD	KDD99
Accuracy	82	80.59	92.5	90	94.3	92	97	96.7	98.5	98
Precision	82.7	79.5	91.9	89.1	95	93.1	96.5	95.3	98.2	97
Recall	82.9	79.9	89.3	88.5	94.9	92.8	97.1	96	99	98
F1-Score	81.4	78.5	90.6	89.6	93	89	96.8	95.8	98	97.9
AUC-Score	82.3	77.4	90	87.8	90	87.6	96	96.9	97.5	97.5

Table 4
Multiclass classification of FEDGAN-IDS using UNSW-NB15.

Metrics	Analysis	Shell-code	Worms	Backdoor	Generic	Reconnaissance	Exploits	Fuzzers	DoS	Normal
Accuracy	94	92	93.8	93	92.8	93	95	97	98	98.7
Precision	95.3	93.5	92.9	94.1	94	96.1	95	96.4	98.8	98.7
Recall	96.3	94.6	93	95.2	95.9	98	96.8	96.8	99	97.8
F1-Score	95.4	93.5	94.6	96.6	96.7	97.5	97.5	97.3	98.7	98.5
AUC-Score	96	94.5	93.3	94.5	95.6	97	98	96.9	98.5	99.4

Table 5
Performance comparison of the proposed models with the existing models.

Model	Architecture	Dataset	Accuracy	Precision	Recall
CGAN-IDS (2021) [54]	CNN	KDD99	93.29	–	–
Bi-GAN IDS (2020) [55]	DNN	KDD99	89.5	83.6	99.4
FEDGAN-IDS (Proposed)	ACGAN and CNN	KDD99	**99.1**	**97.8**	**96.3**
Bi-GAN IDS (2022) [56]	DNN and Encoders	NSL-KDD	91.2	87.27	98.1
GAN-based IDS (2021) [57]	MLP	NSL-KDD	78	78	77
WGAN-IDS (2021) [58]	ANN	NSL-KDD	92	94	93
HFL-IDS (2021) [27]	DNN	NSL-KDD	78.8	76	77
FED-ACNN-IDS (2021) [59]	CNN	NSL-KDD	99	93	92.5
FEDGAN-IDS (Proposed)	ACGAN and CNN	NSL-KDD	**99.29**	**99.3**	**98.9**
CGAN-IDS (2021) [54]	CNN	UNSW-NB15	89.73	–	–
WGAN-IDS (2021) [58]	ANN	UNSW-NB15	83	86	84
FLUIDS (2022) [60]	AutoEncoders and ANN	UNSW-NB15	87	–	–
FL-IDS [61]	ML	UNSW-NB15	85	86.4	83
FL-IDS [62]	DNN	UNSW-NB15	84.3	86.1	83.1
Data privacy FL-IDS (2021) [63]	CNN	UNSW-NB15	82	–	–
FEDGAN-IDS (Proposed)	ACGAN and CNN	UNSW-NB15	**99.4**	**99.56**	**99.3**

We record the performance of multiclass FEDGAN-IDS model for different attack classes. Fig. 13 shows the performance recorded at each global epoch of FEDGAN-IDS modeled as multiclass classifier using NSL-KDD dataset. From the figure, we observe that the performance of GAN-based Federated Multiclass classification IDS model is satisfactory. The data augmentation on each attack class samples improves the accuracy at each global epoch. We observe that for every global epoch, there is approximately 3%–5% increase in the accuracy for each attack class. For instance, initially the accuracy U2R and R2L attack class is recorded as 53% and 61%, respectively. After 9 global epochs the accuracy of these two classes is increased by 15% and 19%, respectively. The detailed results after final global update are shown in Table 3. From table, we notice that the performance of FEDGAN-IDS multiclass classification model using NSL-KDD and KDD99 for various metrics lies between 82%–98.5% and 80%–98%, respectively. Whereas, Table 4 shows the performance with UNSW-NB15 datasets which records between 91.5%–98.7%. Correspondingly, the values in the two tables infer that the FEDGAN-IDS gives comparable performance to the existing models. Since FL is used in modeling this architecture is suitable to incrementally train the model with newer attack categories.

Finally, we have also compared the performance of our proposed model with the existing IDS models using GAN and FL, respectively. The comparison is provided in Table 5. Among the existing methods, we have chose the IDS models tested using the same three datasets that we have used. The IDS models based on GAN have achieved performance ranging between 78 and 93. On the contrary, FL based IDS models have shown higher accuracy when compared to GAN based standalone IDS models, with the values ranging between 82–99. Overall, we have seen that the performance of the proposed model is optimal in all the cases with three datasets.

6. Conclusion

In this article, we propose an efficient IDS using GAN network with a distributed FL model. We have discussed the issues and limitations of current IDS for IoT devices. With a series of evaluation and comparison of various existing and proposed models, we prove that the FEDGAN-IDS architecture for IDS in the binary and multiclass classification scenarios are promising for present-day IoT networking. The application of GAN is used for the first time in the federated scenario for IDS.

The model achieves 99% and 98% accuracy for binary and multiclass classification, respectively.

In future, we plan to improve the following critical areas of the proposed model:

(1) We aim to model this architecture including cloud and multiple Edge nodes by allowing two level aggregation in the hierarchical federated learning architecture.
(2) Secure sharing of local gradients and global updates to ensure the integrity in order to protect the Generator and Discriminator networks training against interception attacks.

CRediT authorship contribution statement

Aliya Tabassum: Literature review, System design, Methodology, Solution investigation, Simulation setup, Performance evaluation, Writing – original draft. **Aiman Erbad:** System design, Methodology, Solution investigation, Writing – review & editing. **Wadha Lebda:** System design, Methodology, Writing – review & editing. **Amr Mohamed:** System design, Methodology, Writing – review & editing. **Mohsen Guizani:** System design, Methodology, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This publication was made possible by NPRP grant 7-1469-1-273 from the Qatar National Research Fund (a member of Qatar Foundation). The findings achieved herein are solely the responsibility of the authors.

References

[1] A. Tabassum, W. Lebda, Security framework for IoT devices against cyber-attacks, 2019, arXiv preprint arXiv:1912.01712.

[2] H. Moustafa, E.M. Schooler, G. Shen, S. Kamath, Remote monitoring and medical devices control in ehealth, in: 2016 IEEE 12th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), IEEE, 2016, pp. 1–8.

[3] B. Li, Y. Wu, J. Song, R. Lu, T. Li, L. Zhao, DeepFed: Federated deep learning for intrusion detection in industrial cyber–physical systems, IEEE Trans. Ind. Inf. 17 (8) (2020) 5615–5624.

[4] A. Drewek-Ossowicka, M. Pietrołaj, J. Rumiński, A survey of neural networks usage for intrusion detection systems, J. Ambient Intell. Humaniz. Comput. 12 (1) (2021) 497–514.

[5] S.-W. Lee, M. Mohammadi, S. Rashidi, A.M. Rahmani, M. Masdari, M. Hosseinzadeh, et al., Towards secure intrusion detection systems using deep learning techniques: Comprehensive analysis and review, J. Netw. Comput. Appl. 187 (2021) 103111.

[6] A. Tabassum, A. Erbad, M. Guizani, A survey on recent approaches in intrusion detection system in IoTs, in: 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC), IEEE, 2019, pp. 1190–1197.

[7] A. Aldweesh, A. Derhab, A.Z. Emam, Deep learning approaches for anomaly-based intrusion detection systems: A survey, taxonomy, and open issues, Knowl.-Based Syst. 189 (2020) 105124.

[8] C. Sun, A. Shrivastava, S. Singh, A. Gupta, Revisiting unreasonable effectiveness of data in deep learning era, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 843–852.

[9] A. Tabasum, Z. Safi, W. AlKhater, A. Shikfa, Cybersecurity issues in implanted medical devices, in: 2018 International Conference on Computer and Applications (ICCA), IEEE, 2018, pp. 1–9.

[10] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečnỳ, S. Mazzocchi, H.B. McMahan, et al., Towards federated learning at scale: System design, 2019, arXiv preprint arXiv:1902.01046.

[11] A. Antoniou, A. Storkey, H. Edwards, Data augmentation generative adversarial networks, 2017, arXiv preprint arXiv:1711.04340.

[12] C. Fan, P. Liu, Federated generative adversarial learning, in: Chinese Conference on Pattern Recognition and Computer Vision (PRCV), Springer, 2020, pp. 3–15.

[13] M. Rasouli, T. Sun, R. Rajagopal, Fedgan: Federated generative adversarial networks for distributed data, 2020, arXiv preprint arXiv:2006.07228.

[14] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, Adv. Neural Inf. Process. Syst. 27 (2014).

[15] L. Lan, L. You, Z. Zhang, Z. Fan, W. Zhao, N. Zeng, Y. Chen, X. Zhou, Generative adversarial networks and its applications in biomedical informatics, Front. Public Health 8 (2020) 164, http://dx.doi.org/10.3389/fpubh.2020.00164.

[16] I. Durugkar, I. Gemp, S. Mahadevan, Generative multi-adversarial networks, 2016, arXiv preprint arXiv:1611.01673.

[17] Q. Hoang, T.D. Nguyen, T. Le, D. Phung, Multi-generator generative adversarial nets, 2017, arXiv preprint arXiv:1708.02556.

[18] M. Li, D.G. Andersen, J.W. Park, A.J. Smola, A. Ahmed, V. Josifovski, J. Long, E.J. Shekita, B.-Y. Su, Scaling distributed machine learning with the parameter server, in: 11th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 14), 2014, pp. 583–598.

[19] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, et al., Large scale distributed deep networks, Adv. Neural Inf. Process. Syst. 25 (2012) 1223–1231.

[20] J. Konečnỳ, H.B. McMahan, F.X. Yu, P. Richtárik, A.T. Suresh, D. Bacon, Federated learning: Strategies for improving communication efficiency, 2016, arXiv preprint arXiv:1610.05492.

[21] Q. Yang, Y. Liu, T. Chen, Y. Tong, Federated machine learning: Concept and applications, ACM Trans. Intell. Syst. Technol. (TIST) 10 (2) (2019) 1–19.

[22] J. Yang, C. Zhou, S. Yang, H. Xu, B. Hu, Anomaly detection based on zone partition for security protection of industrial cyber-physical systems, IEEE Trans. Ind. Electron. 65 (5) (2017) 4257–4267.

[23] A. Tabassum, A. Erbad, A. Mohamed, M. Guizani, Privacy-preserving distributed IDS using incremental learning for IoT health systems, IEEE Access 9 (2021) 14271–14283, http://dx.doi.org/10.1109/ACCESS.2021.3051530.

[24] Y. Otoum, D. Liu, A. Nayak, DL-IDS: a deep learning–based intrusion detection framework for securing IoT, Trans. Emerg. Telecommun. Technol. (2019) e3803.

[25] H.-J. Liao, C.-H.R. Lin, Y.-C. Lin, K.-Y. Tung, Intrusion detection system: A comprehensive review, J. Netw. Comput. Appl. 36 (1) (2013) 16–24.

[26] M. Gajewski, J.M. Batalla, G. Mastorakis, C.X. Mavromoustakis, A distributed IDS architecture model for smart home systems, Cluster Comput. 22 (1) (2019) 1739–1749.

[27] H. Saadat, A. Aboumadi, A. Mohamed, A. Erbad, M. Guizani, Hierarchical federated learning for collaborative IDS in IoT applications, in: 2021 10th Mediterranean Conference on Embedded Computing (MECO), IEEE, 2021 pp. 1–6.

[28] S.A. Rahman, H. Tout, C. Talhi, A. Mourad, Internet of things intrusion detection: Centralized, on-device, or federated learning? IEEE Network 34 (6) (2020) 310–317.

[29] T.D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, A.-R. Sadeghi, Dïot: A federated self-learning anomaly detection system for IoT, in: 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS), IEEE, 2019, pp. 756–767.

[30] Y. Fan, Y. Li, M. Zhan, H. Cui, Y. Zhang, Iotdefender: A federated transfer learning intrusion detection framework for 5g iot, in: 2020 IEEE 14th International Conference on Big Data Science and Engineering (BigDataSE), IEEE, 2020 pp. 88–95.

[31] V. Mothukuri, P. Khare, R.M. Parizi, S. Pouriyeh, A. Dehghantanha, G. Srivastava, Federated learning-based anomaly detection for IoT security attacks, IEEE Internet Things J. (2021).

[32] S. Chen, M. Xue, L. Fan, S. Hao, L. Xu, H. Zhu, B. Li, Automated poisoning attacks and defenses in malware detection systems: An adversarial machine learning approach, Comput. Secur. 73 (2018) 326–344.

[33] D.C. Attota, V. Mothukuri, R.M. Parizi, S. Pouriyeh, An ensemble multi-view federated learning intrusion detection for iot, IEEE Access 9 (2021) 117734–117745.

[34] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, V. Shmatikov, How to backdoor federated learning, in: International Conference on Artificial Intelligence and Statistics, PMLR, 2020, pp. 2938–2948.

[35] T. Yu, E. Bagdasaryan, V. Shmatikov, Salvaging federated learning by local adaptation, 2020, arXiv preprint arXiv:2002.04758.

[36] M. Nasr, R. Shokri, A. Houmansadr, Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning, in: 2019 IEEE Symposium on Security and Privacy (SP), IEEE, 2019, pp. 739–753.

[37] Y. Zhao, J. Chen, J. Zhang, D. Wu, J. Teng, S. Yu, PDGAN: A novel poisoning defense method in federated learning using generative adversarial network, in: International Conference on Algorithms and Architectures for Parallel Processing, Springer, 2019, pp. 595–609.

[38] I. Siniosoglou, P. Sarigiannidis, V. Argyriou, T. Lagkas, S.K. Goudos, M. Poveda, Federated intrusion detection in NG-IoT healthcare systems: An adversarial approach, in: ICC 2021-IEEE International Conference on Communications, IEEE, 2021, pp. 1–6.

[39] E. Seo, H.M. Song, H.K. Kim, Gids: Gan based intrusion detection system for in-vehicle network, in: 2018 16th Annual Conference on Privacy, Security and Trust (PST), IEEE, 2018, pp. 1–6.

[40] A. Ferdowsi, W. Saad, Generative adversarial networks for distributed intrusion detection in the internet of things, in: 2019 IEEE Global Communications Conference (GLOBECOM), IEEE, 2019, pp. 1–6.

[41] Z. Chkirbene, H.B. Abdallah, K. Hassine, R. Hamila, A. Erbad, Data augmentation for intrusion detection and classification in cloud networks, in: 2021 International Wireless Communications and Mobile Computing (IWCMC), IEEE, 2021, pp. 831–836.

[42] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, V. Chandra, Federated learning with non-iid data, 2018, arXiv preprint arXiv:1806.00582.

[43] C. Dwork, A. Roth, et al., The algorithmic foundations of differential privacy., Found. Trends Theor. Comput. Sci. 9 (3–4) (2014) 211–407.

[44] B. Xin, W. Yang, Y. Geng, S. Chen, S. Wang, L. Huang, Private fl-gan: Differential privacy synthetic data generation based on federated learning, in: ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2020, pp. 2927–2931.

[45] C. Hardy, E. Le Merrer, B. Sericola, Md-gan: Multi-discriminator generative adversarial networks for distributed datasets, in: 2019 IEEE International Parallel and Distributed Processing Symposium (IPDPS), IEEE, 2019, pp. 866–877.

[46] R. Vinayakumar, K. Soman, P. Poornachandran, Applying convolutional neural network for network intrusion detection, in: 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), IEEE, 2017, pp. 1222–1228.

[47] M. Tavallaee, E. Bagheri, W. Lu, A.A. Ghorbani, A detailed analysis of the KDD cup 99 data set, in: 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ieee, 2009, pp. 1–6.

[48] S. Revathi, A. Malathi, A detailed analysis on NSL-KDD dataset using various machine learning techniques for intrusion detection, Int. J. Eng. Res. Technol. (IJERT) 2 (12) (2013) 1848–1853.

[49] S.I. Popoola, R. Ande, B. Adebisi, G. Gui, M. Hammoudeh, O. Jogunola, Federated deep learning for zero-day botnet attack detection in IoT edge devices, IEEE Internet Things J. (2021).

[50] N. Moustafa, J. Slay, UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set), in: 2015 Military Communications and Information Systems Conference (MilCIS), 2015, pp. 1–6, http://dx.doi.org/10.1109/MilCIS.2015.7348942.

[51] M. Ahmad, Q. Riaz, M. Zeeshan, H. Tahir, S.A. Haider, M.S. Khan, Intrusion detection in internet of things using supervised machine learning based on application and transport layer features using UNSW-NB15 data-set, EURASIP J. Wireless Commun. Networking 2021 (1) (2021) 1–23.

[52] C.C. Aggarwal, et al., Neural Networks and Deep Learning, 10, Springer, 2018, pp. 438–448.

[53] A. Odena, C. Olah, J. Shlens, Conditional image synthesis with auxiliary classifier gans, in: International Conference on Machine Learning, PMLR, 2017 pp. 2642–2651.

[54] G. Andresini, A. Appice, L. De Rose, D. Malerba, GAN augmentation to deal with imbalance in imaging-based intrusion detection, Future Gener. Comput. Syst. 123 (2021) 108–127.

[55] M.O. Kaplan, S.E. Alptekin, An improved BiGAN based approach for anomaly detection, Procedia Comput. Sci. 176 (2020) 185–194.

[56] W. Xu, J. Jang-Jaccard, T. Liu, F. Sabrina, Training a bidirectional GAN-based one-class classifier for network intrusion detection, 2022, arXiv preprint arXiv:2202.01332.

[57] W. Fu, L. Qian, X. Zhu, GAN-based intrusion detection data enhancement, in: 2021 33rd Chinese Control and Decision Conference (CCDC), IEEE, 2021 pp. 2739–2744.

[58] X. Liu, T. Li, R. Zhang, D. Wu, Y. Liu, Z. Yang, A GAN and feature selection-based oversampling technique for intrusion detection, Secur. Commun. Netw. 2021 (2021).

[59] D. Man, F. Zeng, W. Yang, M. Yu, J. Lv, Y. Wang, Intelligent intrusion detection based on federated learning for edge-assisted internet of things, Secur. Commun. Netw. 2021 (2021).

[60] O. Aouedi, K. Piamrat, G. Muller, K. Singh, FLUIDS: Federated learning with semi-supervised approach for intrusion detection system, in: 2022 IEEE 19th Annual Consumer Communications & Networking Conference (CCNC), IEEE, 2022, pp. 523–524.

[61] M. Sarhan, S. Layeghy, N. Moustafa, M. Portmann, A cyber threat intelligence sharing scheme based on federated learning for network intrusion detection, 2021, arXiv preprint arXiv:2111.02791.

[62] O. Aouedi, K. Piamrat, G. Muller, K. Singh, Intrusion detection for softwarized networks with semi-supervised federated learning, in: ICC 2022-IEEE International Conference on Communications, IEEE, 2022, pp. 1–6.

[63] J. Shi, B. Ge, Y. Liu, Y. Yan, S. Li, Data privacy security guaranteed network intrusion detection system based on federated learning, in: IEEE INFOCOM 2021-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), IEEE, 2021, pp. 1–6.