electronics

MDPI

Article

Federated Learning Augmented Cybersecurity for SDN-Based Aeronautical Communication Network

Muhammad Ali *, Yim-Fun Hu and Jian-Ping Li

Bradford-Renduchintala Centre for Space AI, Faculty of Engineering and Digital Technologies, University of Bradford, Bradford BD7 1DP, UK; yfhu@bradford.ac.uk (Y.-F.H.); jpli@bradford.ac.uk (J.-P.L.)
* Correspondence: mali70@bradford.ac.uk

Abstract: With the requirements of government data protection regulations and industrial concerns regarding data protection and privacy, the security level required for data privacy and protection has increased. This has led researchers to investigate techniques that can train cybersecurity machine learning (ML) models without sharing personal data. Federated Learning (FL) is a newly developed decentralized and distributed ML mechanism that emphasize privacy. In this technique, a learning algorithm is trained without collecting or exchanging sensitive data from distributed client models running at different locations. With the rapid increase in the number of cybersecurity attacks reported in the aviation industry in the last two decades, strong, dynamic, and effective countermeasures are required to protect the aviation industry and air passengers against such attacks, which can most of the time lead to catastrophic situations. This paper proposes and implements an FL model for identifying cyberattacks on a Software Defined Network (SDN)-based aeronautical communication networks. The machine learning model used in the FL architecture is a Deep Neural Network (DNN) model. The publicly available National Security Laboratory–Knowledge Discovery and Datamining (NSL-KDD) dataset was employed to train and validate the proposed FL model. The simulation results illustrated that the FL-based system can accurately and effectively identify potential cybersecurity attacks and minimize the risk of data and service exposure without degrading model performance. A comparison was also made between the FL and non-FL machine learning models. Preliminary results demonstrated that the FL model outperformed the non-FL machine learning approaches. FL reached an accuracy of 96%, compared to 76% and 83% for NFL.

Keywords: federated learning; cybersecurity; aeronautical communication network; software-defined network

Academic Editor: Aryya Gangopadhyay

Received: 5 February 2025
Revised: 5 April 2025
Accepted: 8 April 2025
Published: 10 April 2025

Citation: Ali, M.; Hu, Y.-F.; Li, J.-P. Federated Learning Augmented Cybersecurity for SDN-Based Aeronautical Communication Network. *Electronics* **2025**, *14*, 1535. https://doi.org/10.3390/electronics14081535

1. Introduction

With the rapid increase in the number of cybersecurity attacks reported in the aviation industry in the last two decades, it is clear that strong, dynamic, and effective countermeasures are required to protect against cybersecurity attacks in the aviation industry, which can otherwise lead to catastrophic situations. Some incidents of cybersecurity attacks on the global aviation industry [1] are summarized in Table 1 and classified into Integrity (I), Availability (A) and Confidentiality (C) attacks. EUROCONTROL reported a 530% year-on-year increase in reported incidents [2]. The risks associated with the aviation industry due to vulnerable exposure to adversaries such as criminals, cyber-attackers, and highly technical hackers are increasing abruptly with digital and technological advancements. Similarly, these risks increase when the aviation communication network is evolving to meet the growing demands of the industry.

Table 1. Cybersecurity incidents in aviation [1].

Attack Type	Year	Incident	Location
A	2022	DoS attack choked airline operation	India
A	2022	Check-in software compromised	Canada
A	2022	Server compromised lost 65TB of data	Russia
C	2021	SITA data servers compromised	USA
I	2021	Software error	Birmingham, UK
C	2020	Ransomware attack	San Antonio, USA
C	2020	Ransomware attack	Denver, USA
C	2019	Phishing attack	New Zealand
C	2019	Crypto-mining malware infection	Europe
C	2019	Ransomware attack	Albany, USA
C	2019	Cyber attack	Toulouse, France
A	2019	Bot attack	Ben Gurion airport, Israel
A	2018	Cyber attack	Sweden
C	2018	Ransomware attack	Chicago, USA
C	2018	Data breach	Washington DC, USA
C	2018	Mobile App data breach	Air Canada, Canada
A	2018	Ransomware attack	Bristol, UK
C	2018	Data breach	Delta Air, USA
C	2018	Data breach	British Airways, UK
C	2018	Data breach	Hong Kong, China
I	2016	Phishing attacks	Ho chi Minh, Veitnam
A	2016	Cyber attack	Boryspil, Ukraine

Global aviation systems inherit most of their vulnerabilities from traditional IP networks. When the aviation industry adapts the traditional IP network to evolve into a broader and well-connected Aeronautical Telecommunications Network (ATN), it automatically inherits the associated risks and vulnerabilities of traditional IP networks, which necessitate routers to analyze the destination using IP addresses to distinguish data connection sources. This makes the physical network infrastructure very complex and difficult to manage and fully utilize. SDN is a comparatively new but well-researched network technique that provides hope for overcoming the limitations of traditional network infrastructures. It has been proven to be effective and can be widely used in the future Internet [3–6]. SDN is different from the traditional networks by separating the control and data planes, providing an opportunity to control network applications with central programs. Considering these characteristics, SDN have facilitated the embedding of ML and AI [5] to support network management features such as security, mobility, configuration, fault detection, performance monitoring, and access control.

The large amount of publicly available data in the form of shared datasets also provides a good opportunity to implement all kinds of ML models in various application areas [6]. Many factors have driven the development of collaborative machine learning, such as the availability of big data, computational power, and the development and research of new learning strategies and models. Additionally, the emergence of decentralized and secure blockchain technologies, which can address the question of data governance and

traceability, provides incentive means, e.g., the user receives a reward for data contribution within the network, and provides a back-end framework for leveraging large private datasets from different stakeholders and organizations.

In traditional machine learning models, all data must be sent to a centralized server for training, raising concerns regarding data privacy and confidentiality. To address these issues, federated learning (FL) has been introduced and widely adopted in recent research. FL allows multiple clients or contributors to collaboratively solve machine learning problems while maintaining data privacy. In this approach, the clients store their data locally and do not share or transfer them to a central server. Each client trains its model using locally stored data and then shares only model updates or hyperparameters with a central FL server, which aggregates them into a global model and redistributes it to the clients [7]. This process is known as decentralized, parallel-distributed learning. Compared to centralized learning methods, FL is well-suited for large-scale data, as it enhances learning efficiency, reduces data privacy risks, and improves the security of data transfer by transmitting only model parameters instead of raw data over public networks.

This paper aims to study the cybersecurity issue of an SDN-based aeronautical communication network developed in the European project "Software defined networking architecture augmented with Artificial Intelligence to improve aeronautical communications performance, security and efficiency" (SINAPSE) [8], which aims to provide innovative, smart, and safe data links for aeronautical communication network systems based on an SDN architecture augmented with artificial intelligence.

This paper's primary contribution is to utilize federated learning techniques to provide cybersecurity solutions for SDN-based aeronautical communication networks. To the best of our knowledge, this is the first study to use time federated learning for the security of SDN-based aeronautical communication systems. The novelty of this paper is the proposal that integrates Software Defined Networking, Federated Learning, and the functional architecture of Future Communication Infrastructure (FCI) in avionics communications for cybersecurity can be described as follows:

1.1. Amalgamation of SDN and Federated Learning in Avionics Networks

- SDN offers centralized control and flexible network management, while traditional avionics communication networks are rigid and difficult to update or secure dynamically. Integrating SDN into avionics communication networks allows:
    - Dynamic reconfiguration of communication paths and protocols in response to threats.
    - Real-time monitoring of traffic for anomalies or malicious activities.
    - Enhanced scalability and adaptability to emerging threats.
- Federated Learning addresses the challenges of data privacy and distributed learning. Instead of centralizing data, FL enables aircraft and ground stations to collaboratively train machine learning models while keeping data local, thereby improving:
    - Data privacy by preventing raw data transfer between aircraft and control centers
    - Model robustness through distributed training, as each aircraft can contribute to a collective understanding of cyber threats without sharing sensitive information

1.2. Integration with FCI Architecture

- The Future Communications Infrastructure is a planned architecture designed to support high-speed, secure, and resilient communication in the aerospace sector. By embedding SDN and FL within the FCI framework, this paper presents a novel holistic approach to secure avionics communication, enhancing:

- Interoperability between different communication systems used by aircraft, air traffic control, and ground systems
- Scalability and futureproofing, since the architecture can adapt to the increasing complexity and cybersecurity demands of future avionics networks

1.3. Unique Contribution to Avionics Cybersecurity

- This approach introduces a cyber-resilient avionics network that combines the programmability of SDN and decentralized learning of FL to create an adaptive defense system. This is particularly significant in the high-stakes domain of avionics, where cybersecurity is critical for safety and network reliability is paramount
- It provides a forward-looking solution aligned with next-generation air traffic management systems that require robust, secure, and scalable communications

In summary, the novelty of this study lies in the integration of cutting-edge technologies (SDN and FL) within the FCI architecture to create a cybersecure, scalable, and adaptive avionics communication network. This approach enhances the system's resilience against evolving cyber threats while maintaining data privacy, which is a key concern in federated systems.

The remainder of this article is structured as follows. In Section 2, related works on federated learning are reviewed. SDN-based aeronautical communication networks are introduced in Section 3. Section 4 describes the implementation of the federated learning model for the cybersecurity of an aeronautical communication network. Section 5 presents a few simulation results, and Section 6 concludes with recommendations for future investigations.

2. Related Work

2.1. Federated Learning

The concept of FL was first developed by Google [9,10] in 2016 to estimate text inputs from a very large number of mobile phones and tablet devices. All devices were used as nodes in which the model was trained locally, and the results were sent to a centralized server to update the global model.

Recently, many research papers, such as those in [11–17], have reviewed the advances and challenges of federated learning from different perspectives. Rahman et al. [11] investigated FL architecture, designs, and developments compared with the centralized ML-based systems. Yin et al. [12] presented a systematic survey on privacy leakage risks related to federated learning techniques and identified future research directions. Zhang et al. [13] analyzed the existing research of federated learning in literature from five areas: privacy mechanism, data partitioning, communication architecture, learning model, and system heterogeneity. Li et al. [14] analyzed a federated learning system from six aspects, such as: architecture, motivation of federation, data distribution, data privacy methodology, learning method, communication, and scale of federation. Li et al. [15] studied the evolution path of the federated learning concept and process and reviewed the applications of FL in industrial engineering for guiding future applications. Li et al. [16] briefly overviewed the existing approaches and discussed the characteristics and challenges of FL.

Based on data partitioning, FL is divided (Yang et al. [17]) into three classes: i.e., horizontal federated learning (HFL) [9,10,18–20], which is suitable for datasets with the same set of features and different space sizes; vertical federated learning (VFL) [21–28], which is appropriate for datasets with similar space and differing features among non-competing organizations; and federated transfer learning (FTL) [29–33], which is suitable when datasets have different features and different spaces or only a few similar instances.

2.2. Applications of Federated Learning

Li et al. [15] comprehensively summarized the applications of federated learning. In their paper, a review on a number of FL applications was conducted, which includes mobile devices [34–37], Multi-access Edge Computing (MEC) [38–41], industrial applications on cyber security such as signal interference, sensing, jamming, financial industry, data mining [42–46], and finally, healthcare [47].

There is great potential for the FL mechanism in the healthcare industry because there are a lot of patient data in each medical institute [15]. These electronic medical records (EMR) contain many meaningful and private patient information. Kim et al. [48] are the first attempts to apply tensor factorization models with the combination of federated learning for phenotyping analysis without sharing patient data. Huang et al. [49] proposed a community-based FL (CBFL) approach to estimate death rate and hospital (ICU) stay time for heart disease patients by using distributed EMR. Brisimi et al. [50] developed a federated learning predictive framework, called the iterative cluster primal dual splitting (cPDS) approach, to estimate heart-related stay rate by collaborating with many institutions.

2.3. Cyber Security with Federated Learning

Some review papers on the literature of cybersecurity with federated learning have been published [51]. However, most existing research has focused on specific sets of cybersecurity problems. For example, Mallah et al. [52] applied federated learning techniques to explore falsified information attacks in mobile wireless communication systems for autonomous vehicles. Mothukuri et al. [53] developed an FL-augmented anomaly detection mechanism to detect intrusions and attacks in internet networks. A number of industries are hesitant to use Internet technology due to the frequent cybersecurity attacks on connected networks over the public domain, i.e., the Internet. Alazab et al. [54] reviewed federated learning techniques and summarized some use cases, such as finance risk and object detection. Alazab et al. [54] also pointed out that there are some opportunities and chances for FL in cybersecurity. This is one of the motivations for this paper to apply FL in SND-based aviation cybersecurity.

The authors of [55] proposed FL for privacy preservation and intrusion detection in the SDN architecture using the EdgeIIoTset dataset. Scalability and lack of detailed explanation of the actual architecture were a few of the issues left untapped in this research, as well as the specification of the dataset, i.e., the actual number of samples and features in the dataset for intrusion attacks in an SDN-based architecture, was also not mentioned. Our proposed paper will explain both the adapted FL model approach and the target functional architecture for deployment in detail. Our proposed approach provides an amalgamation of the FL paradigm, intrusion detection, and Future Communications Infrastructure, differentiating it from all other proposed research works. The authors of [56] proposed another research work with very good results using FL to secure SDN-based networks for IoT applications. The work presented has some limitations, as the authors did not consider deep learning models but only shallow learning models such as K-Nearest Neighbor, Decision Tree, and Random Forest. The dataset used was comprehensive, but it only included data from IoT devices. Our proposed approach is focused particularly on the avionics domain with generic IP network data. Our methods use a deep learning model for the baseline model in the FL paradigm, which is more suitable for situations in which large amounts of data are presented to the machine learning model. In [57], the authors proposed yet another ML model with the FL paradigm to take advantage of both SDN and FL for scalability, automation, and data privacy during the training phase of ML. The authors proposed using a Convolutional Neural Network (CNN) as the baseline model without providing a clear rationale for ignoring the fact that other ML models are not resource hungry like CNN and

are more suitable for text-based data, which in this case is being used in the model and training phases.

From the above overview, it is clear that FL is becoming a popular ML technique and has been applied in different areas. This paper aims to fill the gap in the cybersecurity of SDN-based aeronautical communication networks. Although FL can significantly minimize the required data transfer needed for training and proposes a more protected paradigm for data privacy, it also has shortcomings, such as vulnerability to inference attacks. Inference attacks are adversarial algorithms used to leak information from a model regarding its training data [58]. Inference attacks probe a machine learning model with various input data and weigh the output to uncover secret information. Recently, many studies have shown that FL is susceptible to membership inference attacks (MIAs), which can separate training model members from non-members [3,59,60]. Such a source of information breaches could have serious privacy implications. Furthermore, many studies have shown that multiple defensive mechanisms can be implemented to overcome inference attacks, such as the Differential Privacy (DP) [58,61,62] technique, which introduces noise to the client's sensitive data before sharing individual updates with the FL server, or the Secure Multi-party Computation (SMC) technique, which allows encryption [63] of clients' uploaded hyperparameters [60]. Table 2 presents a high-level comparison of our proposed research with pre-existing research published using the amalgamation of FL and SDN.

Table 2. Comparison of the proposed work with the literature.

Research Work	Dataset	SDN	FL	DL	Avionics (Heterogenous) Wireless Networks	Detailed Integration of SDN, FL and Intrusions Detection with Target Architecture
Proposed work	NSL-KDD	✓	✓	✓	✓	✓
[9]	Synthetic	×	✓	×	×	×
[53]	N-BaIoT	×	✓	✓	×	✓
[55]	EdgeIIoTset	✓	✓	✓	×	×
[56]	N-BaIoT	✓	✓	×	×	×
[57]	CICDDoS2019	✓	✓	✓	×	×
[3]	Multiple	✓	×	×	×	×
[60]	Synthetic	×	✓	×	✓	×
[58]	Synthetic	×	✓	×	×	×
[62]	Synthetic	×	✓	×	×	×
[64]	CIFAR-100	×	✓	✓	×	×

Symbol '✓' represents supported and '×' represents not supported.

3. SDN-Based Communication Network Architecture

3.1. SDN-Basic Controller Modules

SDN is a network management approach that addresses the limitations of traditional networks and satisfies the requirements of today's complex networks by decoupling the networks into a control plane (network control logic) and data plane (underlying hardware). As a result, network management with SDN becomes simple. Figure 1 shows a general SDN architecture, which includes a control plane, data plane, and application plane. Specifically:

- The data plane is the lowest layer of the SDN architecture and deals with data based on the configurations from the control plane. Normally, the data plane consists of network infrastructure, such as switches, access points, and routers.

- The control plane uses software to define and manage traffic routing and network topology.
- The application plane is the topmost layer and consists of a variety of software applications with tasks ranging from control to management.

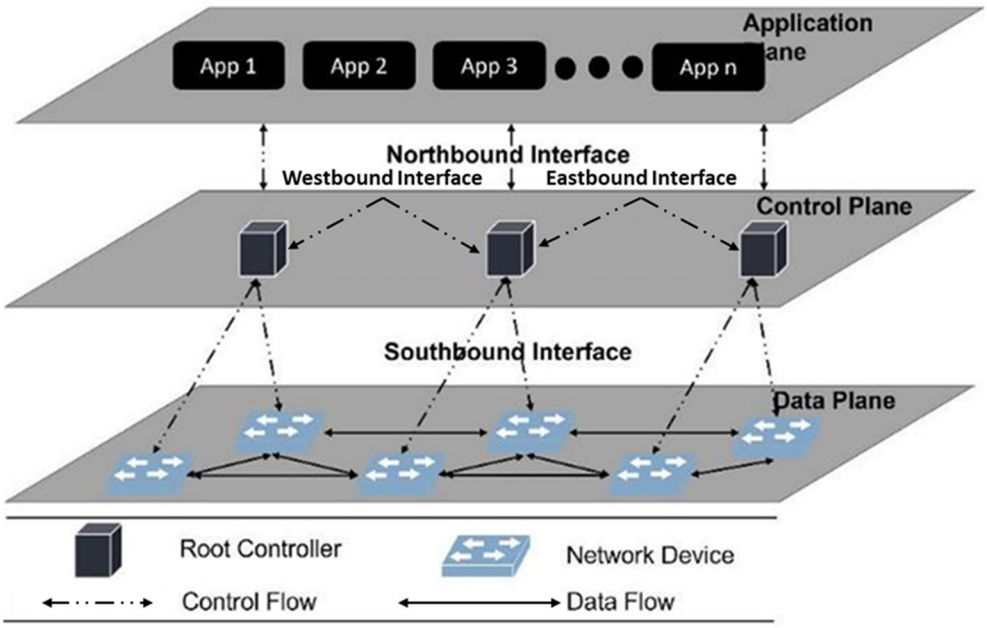

Figure 1. General architecture of software-defined networking.

3.2. Target Network Functional Architecture

The European-funded project SINAPSE [65] aims to improve the performance, security, and efficiency of aeronautical communications by using artificial intelligence (AI) with an SDN architecture.

To reduce the issues of scalability and complexity in an SDN-enabled network, a multi-layered, hybrid, hierarchical control plane structure is proposed, as shown in Figure 2, based on the SDN architecture. There are three administration domains in Control Plane Layer 2 of this architecture. Each domain controller in Layer 2 is connected to the area controllers in Layer 1 within the same administrative domain; each area controller passes control information to the domain controller and, at the same time, controls the devices in the data plane belonging to the same administration domain. Hence, a hierarchical architecture follows within each domain with vertical communications between control layer 2, control layer 1, and the data plane using the north-bound and south-bound interfaces. Communication among different area controllers is performed through the domain controller. Across the different domains, a flat communication architecture across the three domain controllers is assumed using eastbound interfaces. With such a hybrid architecture, each domain controller within the network is up to date on the network state of its neighboring domains, but at the same time maintains a global view of its own domain.

3.3. Target Network Security Functional Architecture

In the SDN-based architecture shown in Figure 2, each layer requires protection against possible adversarial attacks with appropriate security measures for different architectural entities. In the resource layer, the security of networking devices against malicious attacks and security methods for securing communication among SDN controllers and networking devices are important. In the controller layer, communication between controllers and networking devices in the resource layer requires strong security protection. In the application layer, the implementation and facilitation of security features for traffic domains are vital.

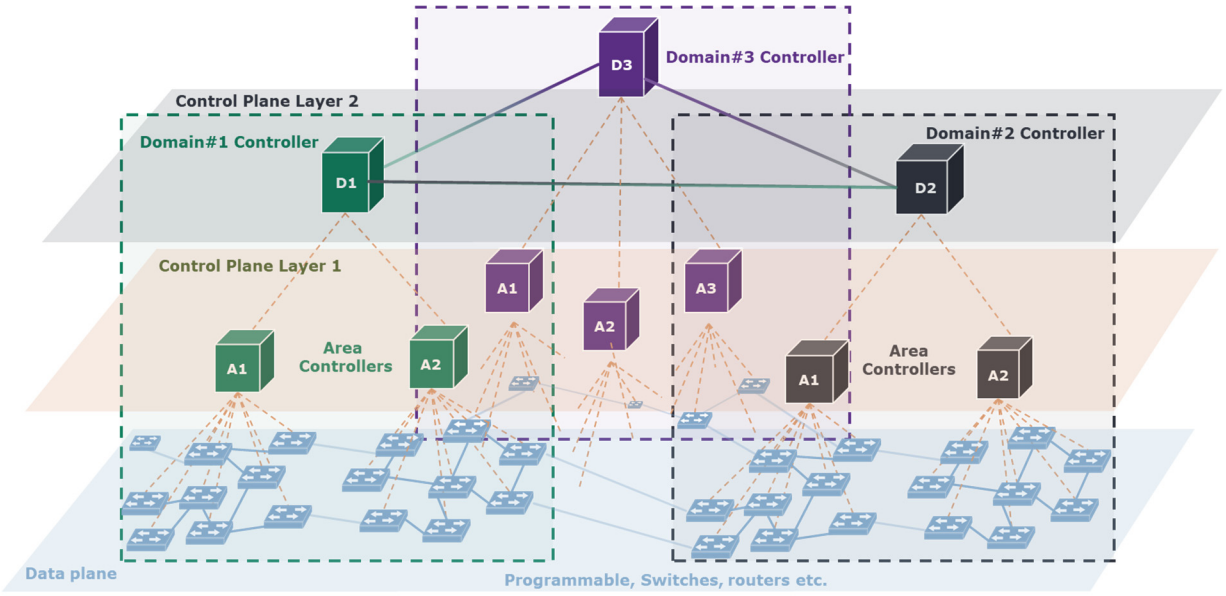

Figure 2. Hybrid SDN controller architecture in target network combining flat and hierarchical models.

Finally, the defense of interfaces between different SDN layers, i.e., north-bound, south-bound, and east/west-bound interfaces, from malicious attacks is a common SDN-specific challenge. Figure 3 presents an overall end-to-end SDN-layered security functional architecture of the target network system with vital security entities highlighted.

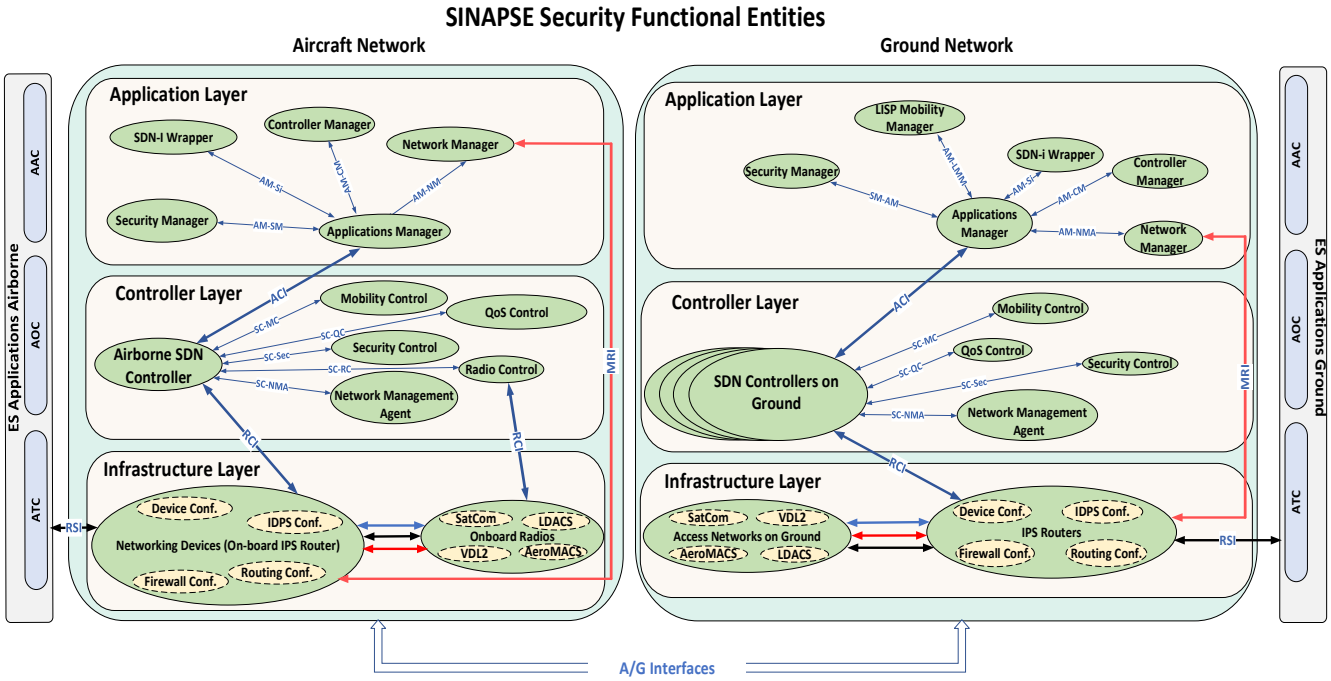

Figure 3. Target network security functional architecture.

4. Federated Learning in Target Network

4.1. Cybersecurity Design Goals for Target Network

The target network architecture aims to develop an ML-augmented network intrusion detection system (NIDS) for binary classifications (0/1), where 0 represents normal traffic and 1 represents attack traffic. NIDS should provide high detection accuracy while preserving data locality and privacy [66] between the domain and area controllers and between

different service provider domains in the target SDN architecture. The design goals of the NIDS are as follows:

- Effectiveness: The NIDS should be able to achieve a high accuracy score in detecting different classes of cyberattacks.
- Privacy Preserving: The NIDS should be able to protect owners' data privacy.
- Scalability: NIDs should be scalable for different cyberattack detection tasks with little modification of the target network system architecture and ML model structures.

These design goals can be achieved using FL. In the target FL architecture context, for any given network administration domain, the domain controllers (DCs) will serve as the FL server, whereas the area controllers (ACs) will act as the clients of the FL server. However, for inter-domain FL implementation, all DCs serve as both clients and servers. For example, if the initial model was initialized on the DC of Domain 1, the DC would maintain the global model and then send the parameters of the initial model to the ACs of Domain 1 through the client-server model and to the neighboring DCs of Domains 2 and 3 as peer entities. The DCs of Domains 2 and 3 will train the obtained model using their own data before passing it to their respective ACs.

4.2. Machine Learning Model

Neural Network Model

In the target network architecture, a Deep Neural Network (DNN) with fully connected layers was chosen as the model for both local and global machine learning tasks.

A DNN was chosen because it is easy to implement, can improve performance with larger datasets, and offers the flexibility to handle classification problems more efficiently. The DNN model designed for implementation consists of five connected hidden layers with sigmoid activation functions. The total number of neurons in the five hidden layers is 30, 20, 10, 5, and 2, respectively, as shown in Figure 4, in which the nodes are numbered sequentially across all layers. The number of neurons in the input and output layers was derived from the dataset input parameters, which were 41 and 1. The number of hidden layers was adapted from the general rules of thumb for the neural network, which are:

- No hidden layers for linearly separable data
- 1–2 hidden layers for less complex data with a low number of input parameters
- 3–5 for the dataset with a high number of input parameters
- More hidden layers can be considered based on the complexity of the dataset in terms of vase features or input parameters, and how comprehensive the dataset is

The number of neurons in each dataset has been selected using the following rules:

- The number of neurons in the hidden layer should be less than twice the input layer
- The number of neurons in the hidden layer should be between the input and output layer neurons

Another rule of thumb adapted for keeping the number of neurons limited in each hidden layer to avoid overfitting is as follows:

$$N_h = \frac{N_s}{(\alpha * (N_i + N_o))} \quad (1)$$

where N_i is the number of input neurons, N_o is the number of output neurons, N_h is the number of neurons in the hidden layer, and α is the scaling factor, which can have values between 2 and 10.

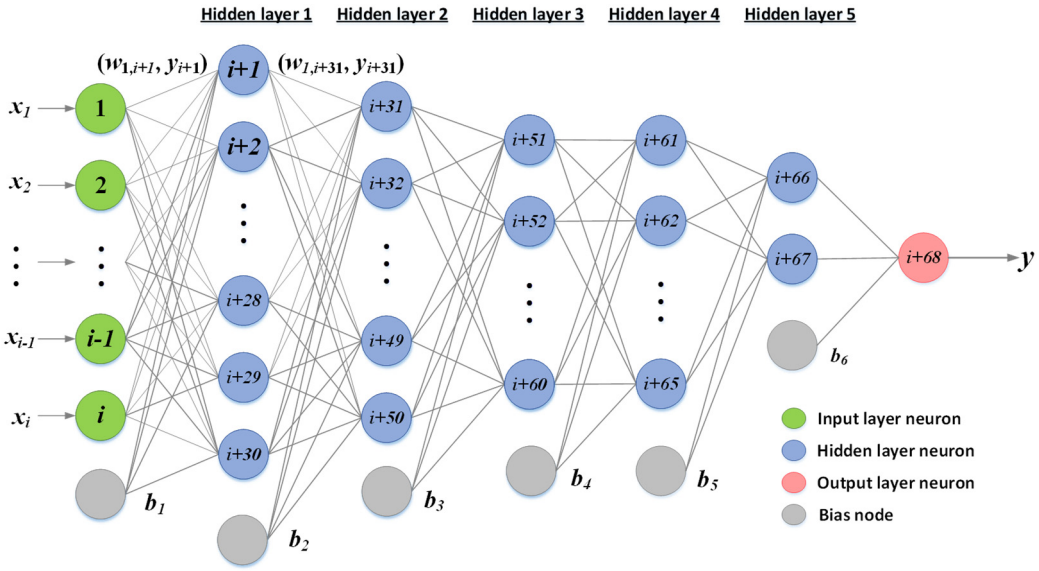

Figure 4. Feed forward back propagate neural network with five hidden layers.

Model training is the most crucial stage of the machine learning process. A supervised machine learning algorithm was chosen for this research, as the dataset contains labeled data. The model uses a sigmoid activation function and a cross-entropy error function for backpropagation during the learning process.

4.3. Federated Learning Models

In FL, there are two main categories of learning models: vertical federated learning and horizontal federated learning. To be more specific, the dataset considered has been divided between multiple aircraft of the same aircraft operator. The horizontal federated learning approach has been used in this study as the dataset used by all the clients has different sample spaces, but the same features.

4.3.1. Horizontal Federated Learning

The datasets of any participating clients, Ci and Cj, have separate sample spaces but the same feature space, such that

$$F_i = F_j, S_i \neq S_j, \mathcal{L}_i \neq \mathcal{L}_j \forall D_i, D_j, i \neq j \quad (2)$$

Figure 5 shows the horizontal federated learning where the Air Navigation Service Provider (ANSP), as the FL server, provides navigation services to *n* aircraft acting as clients. Each aircraft hosts its own local navigation data.

4.3.2. Vertical Federated Learning

Datasets from participating clients have different feature spaces but the same sample space, such that:

$$F_i \neq F_j, S_i = S_j, \mathcal{L}_i \neq \mathcal{L}_j, \forall D_i, D_j, i \neq j \quad (3)$$

Figure 6 shows the vertical data used for federated learning. In the figure, the ANSP and Communications Service Provider (CSP) share a common set of aircraft as their clients (Aircraft 1 and Aircraft *n*), although they provide different services to those aircraft; hence, their service features are different. However, the sample space will be the same since there should not be a one-to-one mapping between the data samples. Since the feature spaces are different, their label spaces will also be different.

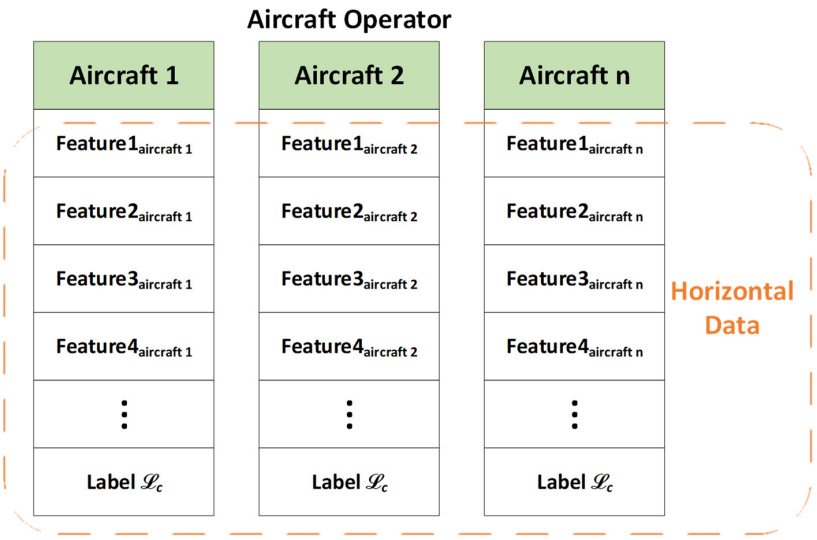

Figure 5. Horizontal federated learning data.

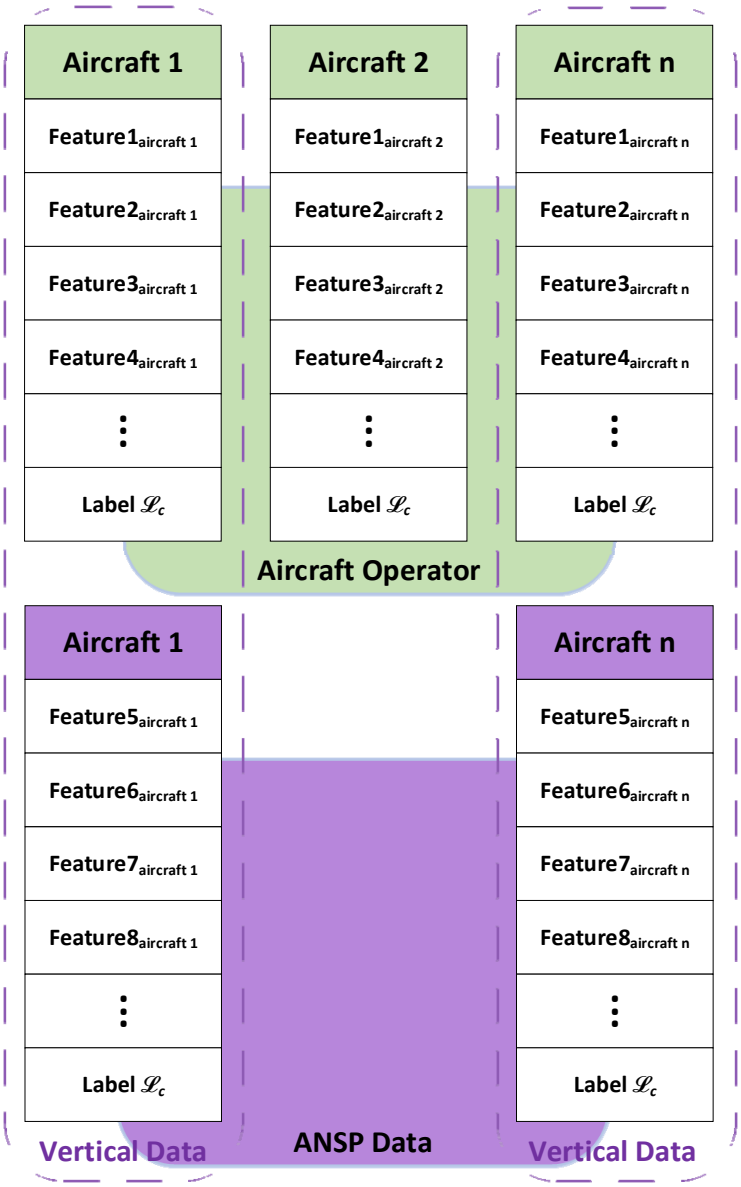

Figure 6. Vertical federated learning data.

4.4. Set up of Federated Learning

4.4.1. FL Training Rounds

In federated learning, the FL server holds and updates the key machine learning model (W). It collects updates from clients on the model parameters and uses this information to refresh the global model. As a result, federated learning involves both local training (on clients) and global training iterations. The training process is represented by discrete time steps, denoted by t ϵ {0, 1, 2, . . .}. The global model is updated after every Te iteration of local training, with the global model typically being updated when $Te = 0$.

$$\boldsymbol{\mathcal{T}} = \{\boldsymbol{t} : \boldsymbol{t\ mod\ T_e} = \boldsymbol{0}\} \tag{4}$$

where $\boldsymbol{\mathcal{T}}$ is the set of times of the global model update.

To start a training ($t\ \epsilon\ \boldsymbol{\mathcal{T}}$), the server should transfer the parameters of the Model $\boldsymbol{W^t} \in \mathbb{R}^d$ of dimension d to all or a small fraction of the set of clients, C_t. Each global training iteration is considered a synchronization step between the server and the clients.

At each global training, the following hyperparameters are exchanged between the client and the server.

- The number of neurons in hidden layer *i*.
- The local weights of client *i* in local training round *t*
- The weights of the global model in training round *t*.
- The total number of hidden layers.
- The learning rates.

4.4.2. FL Training Process

Federated learning involves both training the global model on the server and local training on clients. Stochastic gradient descent (SGD) was applied for training at clients, and the parameters of its model are renewed after each local iteration along the negative gradient direction, which is calculated with random samples from the local dataset. At each global updating step $t\ \epsilon\ \boldsymbol{\mathcal{T}}$, the server transfers the global model $\boldsymbol{W}^{(t)}$ to the clients. Assuming that $w_i^{(t)}$ presents client i's model at time t. When $t\ \epsilon\ \mathcal{T}$, the model $w_i^{(t)}$ is replaced with the global model before a global training process as

$$w_i^{(t)} \leftarrow W^{(t)} \tag{5}$$

Then the local model will be updated by each client $i \in C_t$ within T SGD iterations,

$$\begin{gathered} w_i^{(t+j+1)} = w_i^{(t+j)} - l_{t+j} \nabla F_i\left(w_i^{t+j}, S_i^{t+j}\right) \\ for\ j \in \{0, \ldots, T-1\} \end{gathered} \tag{6}$$

where S_i^{t+j} is a uniformly random sample, $\nabla F_i\left(w_i^{t+j}, S_i^{t+j}\right)$ presents the stochastic gradient, and l_{t+j} is the learning rate (step size).

The stochastic gradient $\nabla F_i\left(w_i^{t+j}, S_i^{t+j}\right)$ is the gradient's unbiased estimator at client i,

$$\mathbb{E}\left[\nabla F_i\left(w_i^{(t+j)}, S_i^{(t+j)}\right)\right] = \nabla F_i\left(w_i^{(t+j)}\right) \tag{7}$$

where $\nabla F_i\left(w_i^{t+j}\right)$ is the gradient of the local loss function.

After Te iterations, client $i \in C_{t+T-1}$ transfers the training results w_i^{t+T} to the global server. For all clients $i \notin C_{t+T-1}$, $w_i^{t+T} = w^{(t)}$.

The server can renew the training model by synchronizing the results obtained from the clients using the Federated Averaging strategy:

$$W^{(t+T)} = \sum_{i \in N} p_i w_i^{(t+T)} \tag{8}$$

After the global model aggregation process is completed, clients will receive the new global model $W^{(t+T)}$ from the server.

4.4.3. Loss Function

Assume that client i has Di data samples, and the total sample size is $D = \sum_{i \in N} D_i$. The objective of federated learning is to train a global ML model, w, and minimize the following global loss function:

$$F(W) = \frac{1}{D} \sum_{i=1}^{N} \sum_{j=1}^{D_i} \mathcal{L}(W,\ x_{ij}) \tag{9}$$

where N is clients total number and $\mathcal{L}(w,\ x_{ij})$ denotes loss at x_{ij} in the local dataset D_i,

The loss function of client i is calculated as

$$F_i(w_i) = \frac{1}{D_i} \sum_{j=1}^{D_i} l(w_i,\ x_{ij}) \tag{10}$$

Substituting Equation (5) into Equation (4), we obtain:

$$F(W) = \sum_{i=1}^{N} p_i F_i(w) \tag{11}$$

where $p_i = \frac{D_i}{D}$ and $\sum_{i \in N} p_i = 1$.

5. Simulation Results

5.1. Datasets

NSL-KDD, which is a refined form of the KDDCup99 dataset is used in this paper for ML model training and evaluation of cyber security applications. NSL-KDD is a well-known cybersecurity dataset that is widely used for benchmarking.

Through simulation, it was shown that using the reduced NSL-KDD dataset with only DDoS attacks and complete NSL-KDD dataset, the learning performance of algorithms becomes superior with increasing number of training samples. Thus, NSL-KDD was used to train and validate the FL model. The traffic features for each training example in the dataset are listed in Table 3. The NSL-KDD dataset includes 41 unique features that can be grouped into three main classes: basic set of features, contents, and traffic features.

Table 3. Traffic features in the NSL-KDD Dataset [67].

#	Feature Name	Feature Description
1	duration	Length of connection in seconds
2	protocol type	Type of protocol i.e., TCP, UDP, ICMP etc.
3	service	Network service on the destination e.g., http
4	flag	Normal or error status of the connection
5	src bytes	Number of data bytes from source to destination
6	dst bytes	Number of data bytes from destination to source
7	land	Represents connection endpoints. 1 if the connection is from same host/port otherwise 0

Table 3. *Cont.*

#	Feature Name	Feature Description
8	wrong fragment	Number of wrong fragments
9	urgent	Number of urgent packets
10	hot	Number of hot indicators
11	num_failed_logins	Number of failed login attempts
12	logged in	1 if successfully logged in, zero otherwise
13	num_compromised	Number of compromised conditions
14	root shell	1 if root shell is obtained, zero otherwise
15	su_attempted	1 if "su root" command is attempted, zero otherwise
16	num root	Number of root access
17	num_file_creations	Number of file creation operation
18	num shells	Number of shell prompts
19	num_access_files	Number of operations on access control files
20	num_outbound_cmds	Number of outbound commands in an ftp connection
21	is_host_login	1 if the login belongs to the hot list otherwise 0
22	is_guest login	1 if the login is a guest login otherwise 0
23	count	Number of connections to the same host
24	srv_count	Number of connection to the same service
25	serror_rate	% of SYN error on the same host connection
26	srv_serror rate	% of SYN error on the same service connection
27	rerror rate	% of REJ error on the same host connection
28	srv_rerror rate	% of REJ error on the same service connection
29	same_srv_rate	Number of same service connected to the same host
30	diff_srv_rate	Number of different services connected to the same host
31	srv_diff_host_rate	Number of different targeted host connected to the same service
32	dst_host_count	Number of connection to same host
33	dst_host_srv_count	Number of same host and same service
34	dst_host_same_srv_rate	Rate of same host and same service
35	dst_host_diff_srv rate	Rate of different service in different host
36	dst_host_same_src_port_rate	Rate of connecting host in same src port
37	dst_host_diff_src_port_rate	Rate of connecting host in different src port
38	dst_host_serror_rate	% of SYN error from the same host connection
39	dst_host_srv_serror_rate	% of SYN error from the same service connection
40	dst_host_rerror_rate	% of REJ error from the same host connection
41	dst_host_srv_rerror_rate	% of REJ error from the same service connection

- *Basic features:* These are TCP/IP-related properties from the packet header, such as service flags and protocol type. In the NSL-KDD dataset, the first nine features are basic.

- *Content features:* These are suspicious data in a TCP packet, such as the number of unsuccessful login attempts. Features 10–22 are grouped as content features in the NSL-KDD dataset.
- *Traffic features:* These are the information related to how the data are transferred to identify the same service and host features. In the NSL-KDD dataset, features 23–31 are classified as service-based features. The remaining features are classified as host-based features.

The NSL-KDD dataset includes four types of attacks: User-to-Root (U2R), Probing (PROB), Denial-of-Service (DoS), and Remote-to-Local (R2L) attacks. The distributed training process encompasses 38 of the 41 features. Feature 6 (dst_bytes), Feature 17 (num_file_creations), and Feature 18 (num_shells) were not considered for training, as these features from the dataset would not be able to efficiently play a vital role in the ML model's learning process because their variance values were NULL.

The reduced NSL-KDD Dataset in DoS attacks was used for training the FL clients and for the initial training at the server. The dataset used for Federated Learning comprises 29K samples in total. In the implementation, only two clients on different aircraft are considered. The dataset is pre-processed and partitioned as follows:

- The 29K samples in the dataset were initially processed to remove any redundant records. The dataset was pre-processed using horizontal FL data division, as the feature space was common among all parties.
- The dataset is grouped as follows:
    - 40% for FL client1 training
    - 40% for FL client2 training
    - 20% for testing and validation

The FL server uses the full dataset being allocated for initial training, whereas the FL clients incrementally use portions of their data segments in different iterations to examine the effect of training on their local ML models over 15 FL training rounds, i.e., 5% of the allocated dataset, then 10%, . . ., 45%, 50%, 60%, . . ., 90%, and 100%.

5.2. Simulation Process

The simulation for this work was carried out using MATLAB R2023b V23.2, where two FL clients and one FL server were implemented, and different segments of data from the dataset were allocated. The specifications of the machine used for this simulation are 32 GB RAM, Intel i7 CPU @ 3.2 GHz, and NVIDIA GeForce GPU with a dedicated memory of 6 GB. Once the clients obtain the initial model from the server, they retrain the local model using the same DNN algorithm and then send the aggregated weight/hyperparameter information to the server. The FL server aggregates all the weights using the average aggregation method. The aggregated weight information is then redistributed to both the FL clients. FL clients continuously train the DNN model and periodically update the weights.

5.3. Results and Discussions

Different performance metrics are used to evaluate the performance of the proposed FL. The results shown in the range of graphs in this section present the cumulative statistics of the performance metrics at client1, client2, and the FL server. These graphs show differences in the performance metric values at each site, i.e., client1, client2, and the FL server, at different times throughout the simulation time. The variation in the performance metric values at different intervals is due to the quantity and caliber of the training data exposed to the model at each location.

5.3.1. Accuracy

Figure 7 illustrates that the algorithm's accuracy increases with an increase in the training iterations. To prevent degradation in the model performance, a condition is imposed on the server side such that the aggregated accuracy of the model must be improved in the last three model states using the accuracy metric. If not, then the server disregards the changes and redistributes the last best model to its clients. In an ATN network, the number of participating clients is low and safety-critical and private; thus, an algorithm with a good overall accuracy is required.

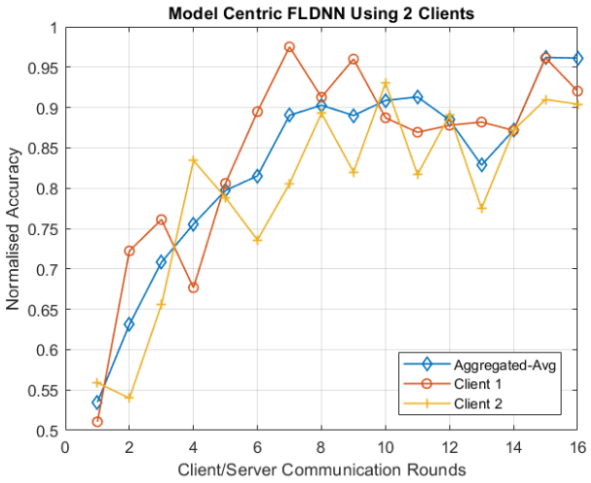

Figure 7. Accuracy graph of FL-DNN for two clients.

However, when the client results are aggregated at the server, the variations in the graphs are considerably reduced. The algorithm consistently improves with an increasing number of communication rounds. Thus, the condition imposed at the server helps the algorithm to only converge and not degrade, and the accuracy graph keeps improving until it becomes a flat line at around 96%, as shown in Figure 7.

5.3.2. Precision

The graph of the precision shown in Figure 8 confirms that as the algorithm started to converge after about 15 rounds of communication, the performance started to flatten, and more than 90% of all positive samples are classified as positive by the algorithm.

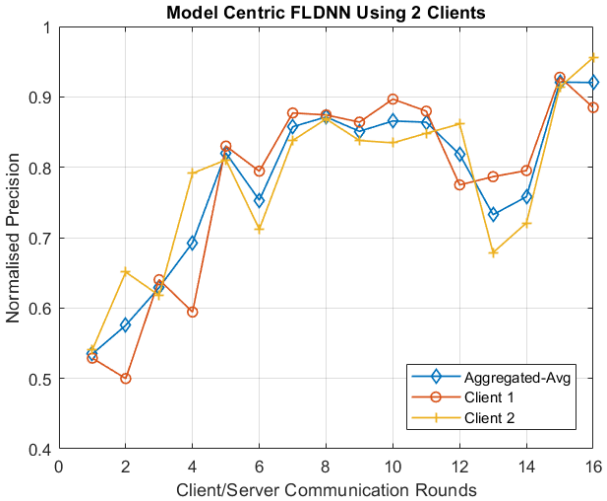

Figure 8. Precision graph of FL-DNN for two clients.

5.3.3. True Positive Rate (TPR)

Figure 9 shows that the ability of the model to detect attacks also improves with an increasing number of training. As evident from the figure, there are times when the aggregated performance decreases, and the ability of the model to detect attacks might deteriorate; selected redistribution ensures that the aggregated but degraded model is not redistributed.

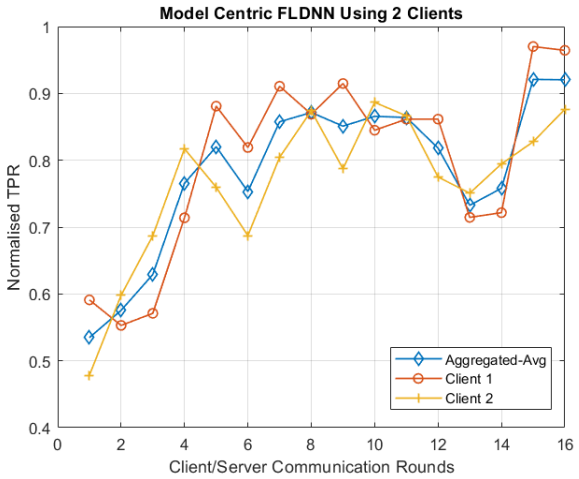

Figure 9. TPR graph of FL-DNN for two clients.

5.3.4. F1-Score

The F1-score or F-measure provides the confidence level in the algorithm's ability for attack detection and is especially useful in the case of cybersecurity. All the results shown in Figure 10 confirm the theoretical assumption that with an increased number of training and client-server communication rounds, both local algorithms' individual performance at the clients and the aggregated average model performance have improved. Figure 10 shows approximately 95% confidence in the model's attack detection performance.

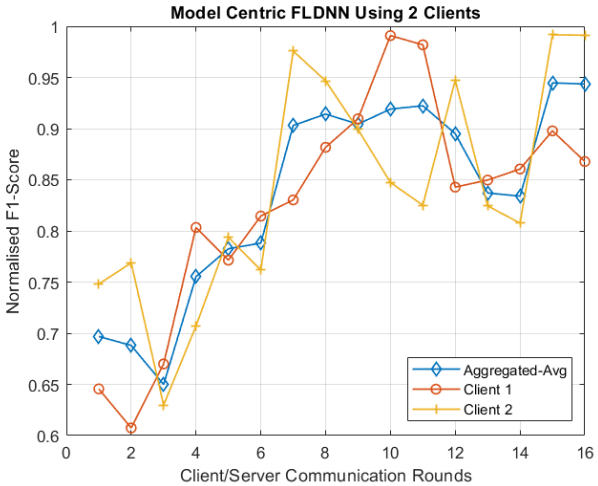

Figure 10. F1/F-measure graph of FL-DNN for two clients.

5.3.5. False Negative Rate, False Positive Rate, and Error Rate

The False Negative Rate (FNR) is defined as the ratio of the number of samples that are incorrectly identified as positive to the total number of samples in a dataset. The False Positive rate (FPR) is defined as the ratio of the number of samples that are incorrectly identified as negative to the total sample number. Finally, the Error rate (ER) is defined as the ratio of the number of samples that are incorrectly identified to the total sample number.

Thus, within a given dataset, the lower the values of the FNR, FPR, and ER, the better the algorithm. Figures 11–13 show the good aggregated FNR, FPR, and ER performance of the FL server.

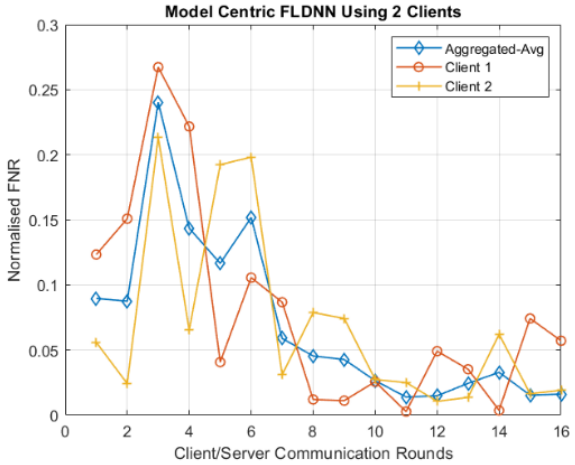

Figure 11. FNR graph of FL-DNN for two clients.

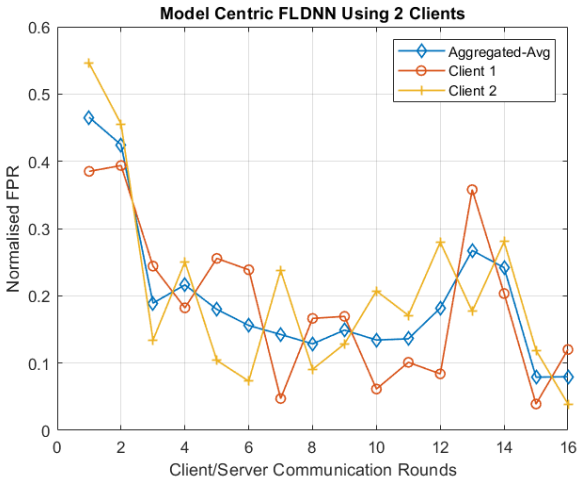

Figure 12. FPR graph of FL-DNN for two clients.

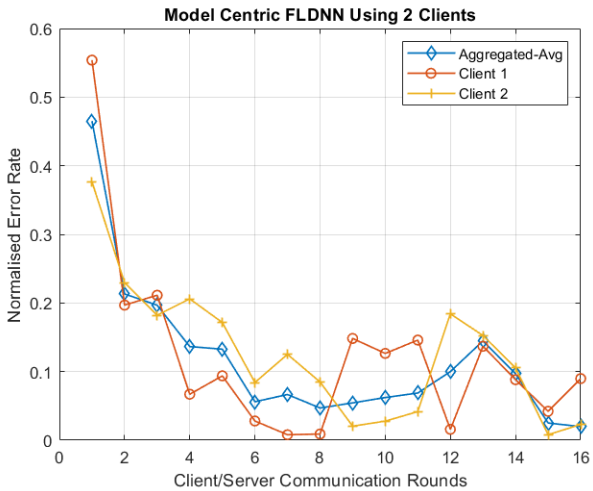

Figure 13. Error rate graph of FL-DNN for two clients.

As the server receives more updates from different clients, the performance continues to improve until these values are below 0.1%. As explained above, for this implementation,

the reduced NSL-KDD dataset with only DoS attacks was employed; thus, the number of training samples was very low for a deep learning algorithm, which is why it took almost 16 communication rounds for the FNR, FPR, and ER values to fall below 0.05%, 0.1%, and 0.05%, respectively. However, large datasets with sufficient training examples and higher local epochs can further improve the performance with reduced communication rounds and will also reduce network traffic.

5.3.6. True Negative Rate (TNR)

The results obtained in Figure 14 provide a high confidence level in the FL server's ability to correctly identify the samples, and this was accomplished relatively quickly within a low number of communication rounds. In addition, it provides a high confidence level in the implementation technique, as the aggregated average is always superior to the weighted average.

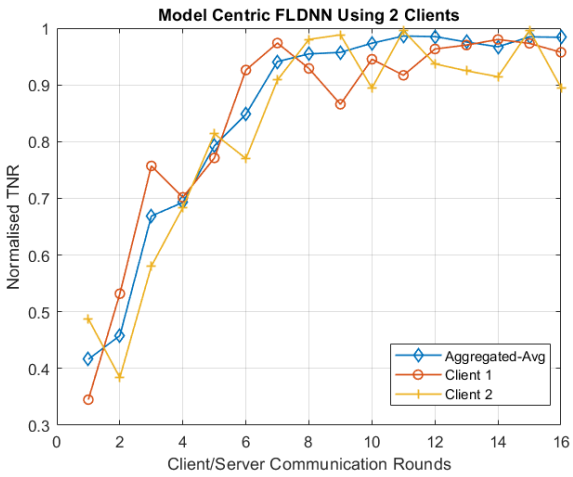

Figure 14. TNR graph of FL-DNN for two clients.

5.4. Comparison with Non-FL Methods

A comparison of the FL and non-FL results has also been carried out by simulating the same distribution of the dataset (NSL-KDD). For a non-FL (NFL) simulation, there is no central server and only two clients, each client receives the initially trained model, then trains the ML model with 40% of the locally available dataset but does not share it with the server.

The results in Figure 15 shows that the accuracy of the FL model is 0.9612l, which is better than those of Client 1 (0.7611) and Client 2 (0.8342). Figure 16 illustrates that the precision of the FL model is 0.9203, which is better than Client 1 (0.6402) and Client 2 (0.7914). The obvious reason is that each client can only train its model using 50% of the data, while the FL uses all the data (100%). The results in Figures 16–21 demonstrate that the proposed FL model outperforms the non-FL model. These results also illustrate that FL could use more data and has the potential to obtain a better model, thereby showing the advantage of FL over the non-FL mechanism of ML training.

The set of graphs from Figures 15–21 shows that FL outperforms the non-FL in all positive and negative performance metrics, whereas the performance between non-FL trained models at two different clients also varies. This variation in performance between the two ML models is due to the difference in the data samples available for each client. In non-FL the clients do not share the model parameters of the updated ML model for aggregation; therefore, the performance of ML models at different clients varies and remains low as compared to the ML model trained using the FL technique.

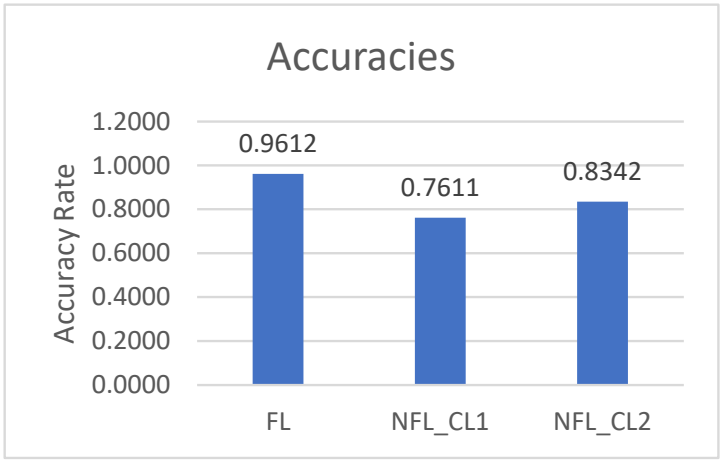

Figure 15. Comparison of FL and Non-FL on Accuracy.

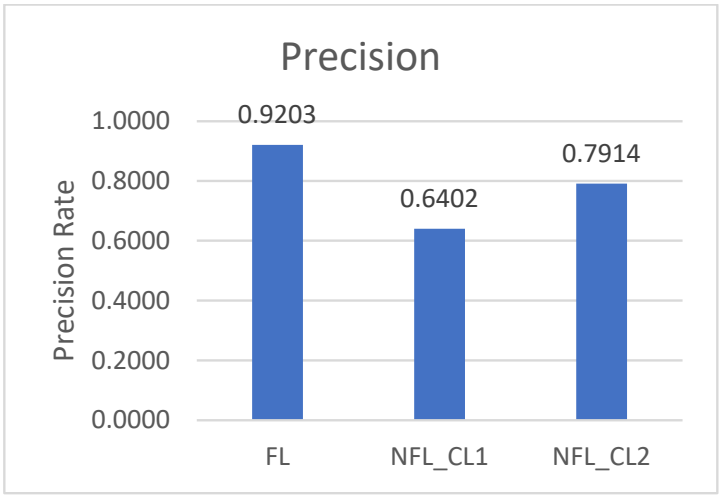

Figure 16. Comparison of FL and non-FL on precision.

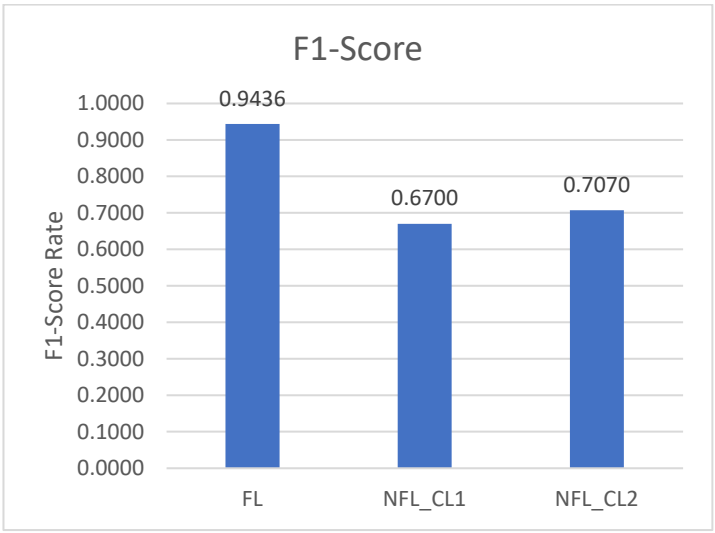

Figure 17. Comparison of FL and non-FL on F1-score.

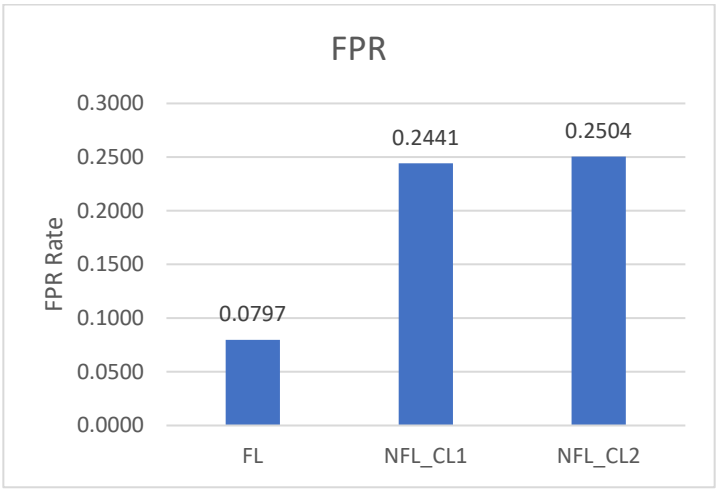

Figure 18. Comparison of FL and non-FL on FPR.

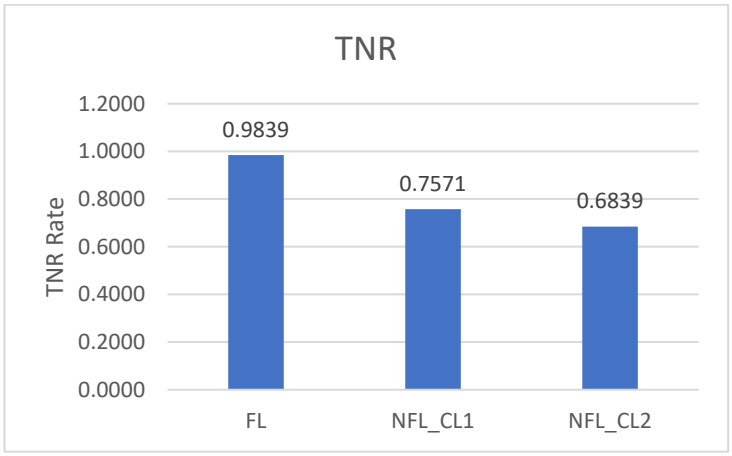

Figure 19. Comparison of FL and non-FL on TNR.

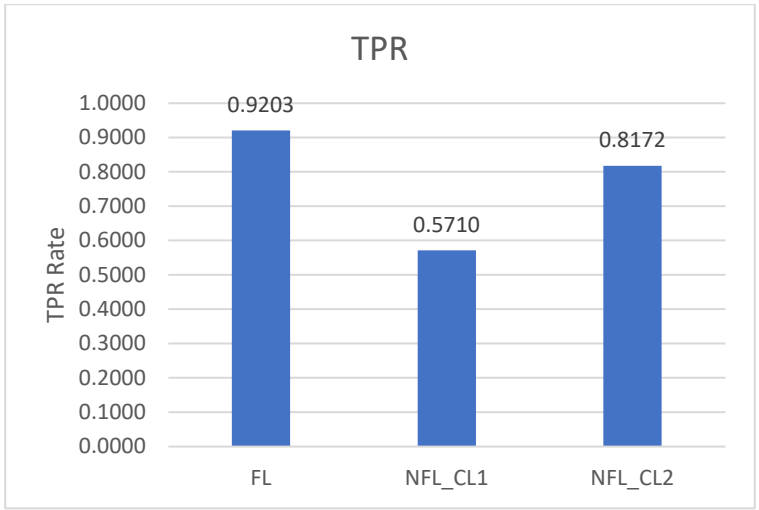

Figure 20. Comparison of FL and non-FL on TPR.

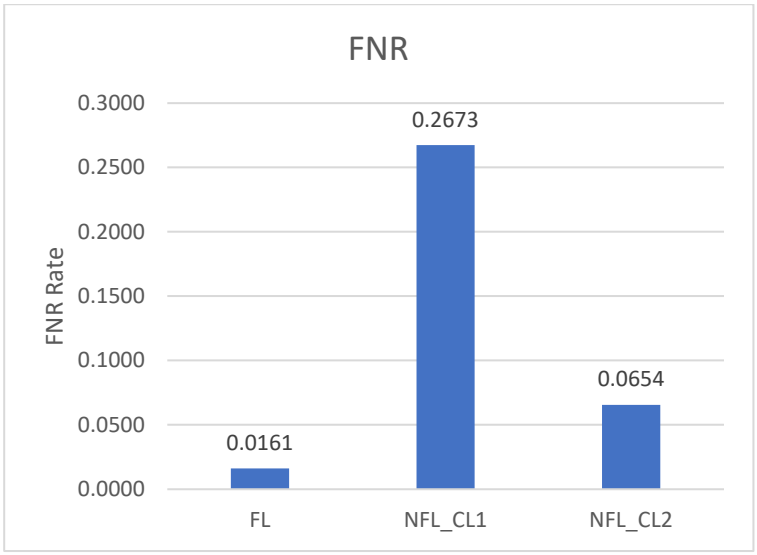

Figure 21. Comparison of FL and non-FL on FNR.

6. Conclusions and Future Works

Horizontal FL is implemented in this paper within a single administration domain, in which the set of features remains the same and the sample spaces are different in the area controllers. The developed FL model was used to validate the cybersecurity attacks on the aeronautical communication system. Domain controllers acting as the FL server for their administration domains initialize model training and send the model hyperparameters to the area controllers. The area controllers update the model locally and immediately implement any updates and send the aggregated model weights to the domain controller. The domain controller will combine the results from all area controllers and send back the aggregated model weights to all area controllers for the next round of training.

The publicly available dataset NSL-KDD was used to simulate the developed model. The records in the dataset were grouped proportionally into different segments for training and validate of the model.

The simulation results demonstrated that federated learning is more feasible for defending the aeronautical communication system against cybersecurity attacks and provides better performance than non-FL models. Although the datasets used for benchmarking the machine learning models are composed of non-avionics IP traffic with attack and non-attack traffic, they can similarly be trained with avionics traffic by applying FL. FL can not only improve data privacy, but also improve the algorithm accuracy for attack detection. The simulation results also demonstrated that the higher the number of iterations of the simulation, the higher the accuracy of the algorithm, until its performance reaches saturation.

Large-scale FL has been addressed in many works. In the future, the pre-developed FL model is aimed to be extended to incorporate large-scale communication networks and address the issues of achieving secure communication between FL clients and the FL server. Further extension of the proposed work aims to deploy the proposed FL technique on a real network and compare its performance with the simulation results. In future work, when considering the deployment and testing of FL-augmented cyber defense for SDN-based aeronautical networks, the model would require benchmarking using a dataset from the avionics domain. AS FL eliminates the need to exchange actual data for training or aggregating the ML model before distributing it to all the FL clients, it still needs to protect the hyperparameters being shared between the FL clients and the FL server. Security can also be another branch of work in future recommendations. Another important factor not

considered in the simulation is the list of constraints, such as network latency, bandwidth, throughput, jitter, and link error or loss rates, that can affect communication between the FL server and FL clients. Last but not least, demonstration of scalability, which has not been considered in this paper due to limited resources, can also be part of future work for this research.

Author Contributions: Conceptualization, M.A.; Software, M.A.; Investigation, M.A.; Writing—original draft, M.A.; Writing—review & editing, M.A., Y.-F.H. and J.-P.L. All authors have read and agreed to the published version of the manuscript.

Funding: The research work presented in this paper was taken from the SINAPSE project, which received funding from the SESAR Joint Undertaking under the European Union's Horizon 2020 research and innovation programme with grant agreement No 892002. The opinions expressed in this work reflect the authors' views only, and SJU shall not be considered liable for them or for any use that may be made of the information contained herein.

Data Availability Statement: Dataset details are contained within the article. Alternatively the dataset is publicly available from: https://www.kaggle.com/datasets/hassan06/nslkdd (accessed on 5 February 2025).

Acknowledgments: The authors acknowledge the inspiration from the SESAR Joint Undertaking (SJU). The authors also acknowledge the efforts made by Viktor Doychinov for proofreading and Rameez for his contribution to the literature review.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ukwandu, E.; Ben-Farah, M.A.; Hindy, H.; Bures, M.; Atkinson, R.; Tachtatzis, C.; Andonovic, I.; Bellekens, X. Cyber-Security challenges in Aviation Industry: A Review of Current and Future Trends. *Information* **2022**, *13*, 146. [CrossRef]
2. Eurocontrol EATM CERT Services. Aviation Under Attack: Faced with a Rising Tide of Cybercrime, Is Our Industry Resilient Enough to Cope. Think Paper #12. 5 July 2021. Available online: https://www.eurocontrol.int/sites/default/files/2021-07/eurocontrol-think-paper-12-aviation-under-cyber-attack.pdf (accessed on 3 April 2025).
3. Xia, W.; Wen, Y.; Foh, C.H.; Niyato, D.; Xie, H. A Survey on Software-Defined Networking. *IEEE Commun. Surv. Tutor.* **2015**, *17*, 27–51. [CrossRef]
4. Benson, T.; Akella, A.; Maltz, D. Unraveling the complexity of network management. In Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation, Boston, MA, USA, 22–24 April 2009; pp. 335–348.
5. Zhao, Y.; Li, Y.; Zhang, X.; Geng, G.; Zhang, W.; Sun, Y. A Survey of Networking Applications Applying the Software Defined Networking Concept Based on Machine Learning. *IEEE Access* **2019**, *7*, 95397–95417. [CrossRef]
6. Konečný, J.; McMahan, H.B.; Yu, F.X.; Richtárik, P.; Suresh, A.T.; Bacon, D. Federated learning: Strategies for improving communication efficiency. In Proceedings of the NIPS Workshop Private Multi-Party Machine Learning for Complex Systems, Barcelona, Spain, 9 December 2016; pp. 1–10. [CrossRef]
7. Sarker, I.H.; Kayes, A.S.M.; Badsha, S.; Alqahtani, H.; Watters, P.; Ng, A. Cybersecurity data science: An overview from machine learning perspective. *J. Big Data* **2020**, *7*, 1–29. [CrossRef]
8. Hu, Y.-F.; Abdo, K.; BenSlama, F.; Ali, M.; Cormbe, Q.; Benamrane, F.; Luong, D.; Barossi, R.; Cormbe, O. A SDN-based Aeronautical Communications Network Architecture. In Proceedings of the 2018 IEEE/AIAA 37th Digital Avionics Systems Conference (DASC), London, UK, 23–27 September 2018; pp. 1–10. [CrossRef]
9. Konecn, J.; McMahan, H.B.; Ramage, D.; Richtárik, P. Federated optimization: Distributed optimization beyond the datacenter. *arXiv* **2015**, arXiv:1610.02527. [CrossRef]
10. McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; y Arcas, B.A. Communication-Efficient Learning of Deep Networks from Decentralized Data. *arXiv* **2016**, arXiv:1602.05629.
11. Abdulrahman, S.; Tout, H.; Ould-Slimane, H.; Mourad, A.; Talhi, C.; Guizani, M. A survey on federated learning: The journey from centralized to distributed on-site learning and beyond. *IEEE Internet Things J.* **2021**, *8*, 5476–5497. [CrossRef]
12. Yin, X.; Zhu, Y.; Hu, J. A comprehensive survey of privacy-preserving federated learning: A taxonomy, review, and future directions. *ACM Comput. Surv.* **2022**, *54*, 1–36. [CrossRef]
13. Zhang, C.; Xie, Y.; Bai, H.; Yu, B.; Li, W.; Gao, Y. A survey on federated learning. *Knowl.-Based Syst.* **2021**, *216*, 106775. [CrossRef]

14. Li, Q.; Wen, Z.; Wu, Z.; Hu, S.; Wang, N.; Li, Y.; Liu, X.; He, B. A survey on federated learning systems: Vision, hype and reality for data privacy and protection. *IEEE Trans. Knowl. Data Eng.* **2021**, *35*, 3347–3366. [CrossRef]
15. Li, L.; Fan, Y.; Tse, M.; Lin, K.-Y. A review of applications in federated learning. *Comput. Ind. Eng.* **2020**, *149*, 106854. [CrossRef]
16. Li, T.; Sahu, A.K.; Talwalkar, A.; Smith, V. Federated learning: Challenges, methods, and future directions. *IEEE Signal Process. Mag.* **2020**, *37*, 50–60. [CrossRef]
17. Yang, Q.; Liu, Y.; Chen, T.; Tong, Y. Federated machine learning: Concept and applications. *ACM Trans. Intell. Syst. Technol.* **2019**, *10*, 1–19. [CrossRef]
18. Li, Q.; Wen, Z.; He, B. Practical federated gradient boosting decision trees. *Proc. AAAI Conf. Artif. Intell.* **2020**, *34*, 4642–4649. [CrossRef]
19. Smith, V.; Chiang, C.-K.; Sanjabi, M.; Talwalkar, A.S. Federated multi-task learning. In Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017. Available online: http://papers.nips.cc/paper/7029-federated-multi-task-learning.pdf (accessed on 3 April 2025).
20. Lin, Y.; Han, S.; Mao, H.; Wang, Y.; Dally, W.J. Deep gradient compression: Reducing the communication bandwidth for distributed training. In Proceedings of the Sixth International Conference on Learning Representations, Vancouver, BC, Canda, 30 April–3 May 2018. [CrossRef]
21. Liu, Y.; Kang, Y.; Xing, C.; Chen, T.; Yang, Q. Secure federated transfer learning. *arXiv* **2018**, arXiv:1812.03337.
22. Li, T.; Sahu, A.K.; Zaheer, M.; Sanjabi, M.; Talwalkar, A.; Smith, V. Federated optimization in heterogeneous networks. *arXiv* **2018**, arXiv:1812.06127.
23. Cheng, K.; Fan, T.; Jin, Y.; Liu, Y.; Chen, T.; Papadopoulos, D.; Yang, Q. Secureboost: A lossless federated learning framework. *arXiv* **2019**, arXiv:1901.08755.
24. Yin, F.; Lin, Z.; Kong, Q.; Xu, Y.; Li, D.; Theodoridis, S.; Cui, S.R. FedLoc: Federated Learning Framework for Data-Driven Cooperative Localization and Location Data Processing. *IEEE Open J. Signal Process.* **2020**, *1*, 187–215. [CrossRef]
25. Du, W.; Han, Y.S.; Chen, S. Privacy-preserving multivariate statistical analysis: Linear regression and classification. In Proceedings of the 2004 SIAM International Conference on Data Mining (SDM), Lake Buena Vista, FL, USA, 22–24 April 2004. [CrossRef]
26. Vaidya, J.; Clifton, C. Privacy preserving association rule mining in vertically partitioned data. In Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'02), Edmonton, AB, Canada, 23–26 July 2002; ACM: New York, NY, USA, 2002; pp. 639–644. [CrossRef]
27. Wan, L.; Ng, W.K.; Han, S.; Lee, V.C.S. Privacy-preservation for gradient descent methods. In Proceedings of the 13th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD'07), San Jose, CA, USA, 12–15 August 2007; ACM: New York, NY, USA, 2007; pp. 775–783. [CrossRef]
28. Gascón, A.; Schoppmann, P.; Balle, B.; Raykova, M.; Doerner, J.; Zahur, S.; Evans, D. Privacy-preserving distributed linear regression on high-dimensional data. *Proc. Priv. Enhancing Technol.* **2017**, *2017*, 345–364. [CrossRef]
29. Saha, S.; Ahmad, T. Federated Transfer Learning: Concept and applications. *arXiv* **2020**, arXiv:2010.15561.
30. Yang, H.; He, H.; Zhang, W.; Cao, X. FedSteg: A federated transfer learning framework for secure image steganalysis. *IEEE Trans. Netw. Sci. Eng.* **2021**, *8*, 1084–1094. [CrossRef]
31. Gao, D.; Liu, Y.; Huang, A.; Ju, C.; Yu, H.; Yang, Q. Privacy-preserving heterogeneous federated transfer learning. In Proceedings of the 2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, 9–12 December 2019; pp. 2552–2559. [CrossRef]
32. Sharma, S.; Xing, C.; Liu, Y.; Kang, Y. Secure and efficient federated transfer learning. In Proceedings of the 2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, 9–12 December 2019; pp. 2569–2576. [CrossRef]
33. Pan, S.J.; Yang, Q. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.* **2010**, *22*, 1345–1359. [CrossRef]
34. Yang, T.; Andrew, G.; Eichner, H.; Sun, H.; Li, W.; Kong, N.; Beaufays, F. Applied federated learning: Improving google keyboard query suggestions. *arXiv* **2018**, arXiv:1812.02903.
35. Chen, M.; Mathews, R.; Ouyang, T.; Beaufays, F. Federated learning of out-of-vocabulary words. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 4102–4112. [CrossRef]
36. Leroy, D.; Coucke, A.; Lavril, T.; Gisselbrecht, T.; Dureau, J. Federated learning for keyword spotting. In Proceedings of the ICASSP 2019–2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, UK, 12–17 May 2019; pp. 6341–6345. [CrossRef]
37. Ramaswamy, S.; Mathews, R.; Rao, K.; Beaufays, F. Federated learning for emoji prediction in a mobile keyboard. *arXiv* **2019**, arXiv:1906.04329.
38. Zaw, C.W.; Pandey, S.R.; Kim, K.; Hong, C.S. Energy-aware resource management for federated learning in multi-access edge computing systems. *IEEE Access* **2021**, *9*, 34938–34950. [CrossRef]
39. Wang, S.; Tuor, T.; Salonidis, T.; Leung, K.K.; Makaya, C.; He, T.; Chan, K. Adaptive federated learning in resource constrained edge computing systems. *IEEE J. Sel. Areas Commun.* **2019**, *37*, 1205–1221. [CrossRef]

40. Yang, Z.; Chen, M.; Wong, K.-K.; Poor, H.V.; Cui, S. Federated Learning for 6G: Applications, Challenges, and Opportunities. *Engineering* **2022**, *8*, 33–41. [CrossRef]
41. Hu, B.; Gao, Y.; Liu, L.; Ma, H. Federated region-learning: An edge computing based framework for urban environment sensing. In Proceedings of the 2018 IEEE Global Communications Conference (GLOBECOM), Abu Dhabi, United Arab Emirates, 9–13 December 2018; pp. 1–7. [CrossRef]
42. Han, X.; Yu, H.; Gu, H. Visual inspection with federated learning. In Proceedings of the International Conference on Image Analysis and Recognition (ICIAR 2019), Waterloo, ON, Canada, 27–29 August 2019; pp. 52–64. [CrossRef]
43. Mowla, N.I.; Tran, N.H.; Doh, I.; Chae, K. Federated learning-based cognitive detection of jamming attack in flying Ad-Hoc network. *IEEE Access* **2019**, *8*, 4338–4350. [CrossRef]
44. Saputra, Y.M.; Hoang, D.T.; Nguyen, D.N.; Dutkiewicz, E.; Mueck, M.D.; Srikanteswara, S. Energy demand prediction with federated learning for electric vehicle networks. In Proceedings of the 2019 IEEE Global Communications Conference (GLOBECOM), Waikoloa, HI, USA, 9–13 December 2019; pp. 1–6. [CrossRef]
45. Suvarna, R.; Kowshaly, A.M. Credit card fraud detection using federated learning techniques. *Int. J. Sci. Res. Sci. Eng. Technol.* **2020**, *7*, 356–367. [CrossRef]
46. Yu, B.; Mao, W.; Lv, Y.; Zhang, C.; Xie, Y. A survey on federated learning in data mining. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2022**, *12*, e1443. [CrossRef]
47. Nguyen, D.C.; Pham, Q.-V.; Pathirana, P.N.; Ding, M.; Seneviratne, A.; Lin, Z.; Dobre, O.; Hwang, W.-J. Federated learning for smart healthcare: A survey. *ACM Comput. Surv.* **2023**, *55*, 1–37. [CrossRef]
48. Kim, Y.; Sun, J.; Yu, H.; Jiang, X. Federated tensor factorization for computational phenotyping. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, 13–17 August 2017; pp. 887–895. [CrossRef]
49. Huang, L.; Shea, A.L.; Qian, H.; Masurkar, A.; Deng, H.; Liu, D. Patient clustering improves efficiency of federated machine learning to predict mortality and hospital stay time using distributed electronic medical records. *J. Biomed. Inform.* **2019**, *99*, 103291. [CrossRef] [PubMed]
50. Theodora, S.B.; Chen, R.; Theofanie, M.; Olshevsky, A.; Paschalidis, I.C.; Shi, W. Federated learning of predictive models from federated electronic health records. *Int. J. Med. Inform.* **2018**, *112*, 59–67. [CrossRef]
51. Ghimire, B.; Rawat, D.B. Recent advances on federated learning for cybersecurity and cybersecurity for federated learning for internet of things. *IEEE Internet Things J.* **2022**, *9*, 8229–8249. [CrossRef]
52. Al Mallah, R.; Badu-Marfo, G.; Farooq, B. Cybersecurity threats in connected and automated vehicles based federated learning systems. In Proceedings of the 2021 IEEE Intelligent Vehicles Symposium Workshops (IV Workshops), Nagoya, Japan, 11–17 July 2021; pp. 13–18. [CrossRef]
53. Mothukuri, V.; Khare, P.; Parizi, R.M.; Pouriyeh, S.; Dehghantanha, A.; Srivastava, G. Federated-Learning-Based Anomaly Detection for IoT Security Attacks. *IEEE Internet Things J.* **2022**, *9*, 2545–2554. [CrossRef]
54. Alazab, M.; RM, S.P.; M, P.; Maddikunta, P.K.R.; Gadekallu, T.R.; Pham, Q.-V. Federated Learning for Cybersecurity: Concepts, Challenges, and Future Directions. *IEEE Trans. Ind. Inform.* **2022**, *18*, 3501–3509. [CrossRef]
55. Raza, M.; Saeed, M.J.; Riaz, M.B.; Sattar, M.A. Federated Learning for Privacy-Preserving Intrusion Detection in Software-Defined Networks. *IEEE Access* **2024**, *12*, 69551–69567. [CrossRef]
56. Babbar, H.; Rani, S.; Singh, A.; Gianini, G. Detecting Cyberattacks to Federated Learning on Software-Defined Networks. In Proceedings of the International Conference on Management of Digital, Heraklion, Greece, 5–7 May 2023; Springer Nature: Cham, Switzerland, 2023.
57. Duy, P.T.; Van Hung, T.; Ha, N.H.; Hoang, H.D.; Pham, V.-H. Federated learning-based intrusion detection in SDN-enabled IIoT networks. In Proceedings of the 2021 8th NAFOSTED Conference on Information and Computer Science (NICS), Hanoi, Vietnam, 21–22 December 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 424–429.
58. Benmalek, M.; Benrekia, M.A.; Challal, Y. Security of Federated Learning: Attacks, Defensive Mechanisms, and Challenges. *Rev. D'intelligence Artif.* **2022**, *36*, 49–59. [CrossRef]
59. Luo, X.; Qin, W.; Dong, A.; Sedraoui, K.; Zhou, M. Efficient and High-quality Recommendations via Momentum-incorporated Parallel Stochastic Gradient Descent-Based Learning. *IEEE/CAA J. Autom. Sin.* **2020**, *8*, 402–411. [CrossRef]
60. AICHAIN SESAR Joint Undertaking Project—A Platform for Privacy-Preserving Federated Machine Learning Using Blockchain to Enable Operational Improvements in ATM, 2020/2022. Available online: https://cordis.europa.eu/project/id/894162/reporting (accessed on 3 April 2025).
61. Hu, H.; Salcic, Z.; Sun, L.; Dobbie, G.; Zhang, X. Source Inference Attacks in Federated Learning. In Proceedings of the 2021 IEEE International Conference on Data Mining (ICDM), Auckland, New Zealand, 7–10 December 2021; pp. 1102–1107. [CrossRef]
62. Canetti, R.; Feige, U.; Goldreich, O.; Naor, M. Adaptively secure multi-party computation. In Proceedings of the Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, Philadelphia, PA, USA, 22–24 May 1996; pp. 639–648.

63. Dwork, C.; McSherry, F.; Nissim, K.; Smith, A. Calibrating noise to sensitivity in private data analysis. In Proceedings of the Theory of Cryptography Conference, New York, NY, USA, 4–7 March 2006; pp. 265–284. [CrossRef]
64. Dwork, C. Differential privacy: A survey of results. In Proceedings of the International Conference on Theory and Applications of Models of Computation, Xi'an, China, 25–29 April 2008; pp. 1–19. [CrossRef]
65. SINAPSE Project–Sesar Joint Undertaking. Available online: https://www.sesarju.eu/projects/sinapse (accessed on 4 April 2025).
66. Dwork, C.; Roth, A. *The Algorithmic Foundations of Differential Privacy*; Now Publishers Inc.: Hanover, MA, USA, 2014.
67. NSL-KDD Dataset on Kaggle. Available online: https://www.kaggle.com/code/eneskosar19/intrusion-detection-system-nsl-kdd (accessed on 4 April 2025).