Carbon-aware and Privacy-preserving Federated Anomaly Detection for Containers

Gökcan Cantali*§, Gürkan Gür*, and Burkhard Stiller§
* Zürich University of Applied Sciences (ZHAW) InIT, CH-8401 Winterthur, Switzerland
§ Communication Systems Group (CSG), University of Zürich (UZH), CH-8050 Zürich, Switzerland
canl@zhaw.ch, gueu@zhaw.ch, stiller@ifi.uzh.ch

***Abstract*—Recent advancements in cloud and containerization technologies have motivated application developers to rely more heavily on container orchestration platforms, such as Kubernetes, for hosting applications, necessitating sophisticated defense mechanisms for such container-oriented platforms. Moreover, Federated Learning (FL) has gained traction amid growing concerns about data privacy in Machine Learning (ML) processes, especially for anomaly or intrusion detection systems operating in multi-tenant environments. This paper applies FL to an existing anomaly detection model, namely GATAKU, and incorporates differential privacy to yield a robust and privacy-preserving anomaly detection mechanism for containerized environments. Furthermore, the carbon footprint of the FL pipeline is measured and reduced through smart client selection mechanisms, addressing the rising carbon-emission concerns in ML operations. Experiments conducted on a simulated environment demonstrate that the federated anomaly detection model preserves the model performance while enhancing privacy and sustainability.**

***Index Terms*—federated learning, privacy-preservation, sustainability, anomaly detection, cloud computing.**

I. Introduction

With the emergence of cloud-native software technologies and deployment paradigms, application development has shifted to the cloud environment. This progress has been accompanied by advances in container-level virtualization technologies, such as Docker [1], which have necessitated the use of orchestration platforms to manage container-based software deployments. Among these platforms, Kubernetes is getting increasingly adopted, including the telecommunications industry [2], for hosting containerized applications. On the other hand, such popularity inevitably draws the attention of malicious parties, who may successfully infect a container and attempt to infiltrate the other containers within the same cluster. Hence, it is critical to develop efficient anomaly detection techniques not only for external network traffic but also for inter-pod communication flows in a Kubernetes cluster.

In today's technological landscape, the adoption of Machine Learning (ML) and Artificial Intelligence (AI) techniques is becoming ubiquitous, driven by cheaper hardware, algorithmic advancements, and growing volumes of accessible data. However, the use of such personal data inevitably heightens users' privacy concerns, which has become even worse since the emergence of large language models [3]. These concerns also apply to entities that wish to prevent their sensitive data from leaking to unauthorized parties, a situation that could incur severe financial or legal costs. For instance, in a multi-node Kubernetes cluster, each node may wish to keep its network data confidential and, thus, might be hesitant to share potentially sensitive information, even for the sake of a more accurate ML model that could further enhance security. Similar scenarios are very likely in multi-tenant containerized environments where different operators run their services on the same infrastructure, necessitating proper isolation [4].

Due to traditional ML methods falling short in the privacy aspect, Federated Learning (FL) [5] has been developed with a focus on privacy, aiming to achieve accurate ML models without sharing sensitive data with third parties. The main idea of this approach is to have each data owner train a local model using their own hardware and data, and then collaboratively share the resulting model parameters for aggregation. After its invention, FL quickly became popular across multiple domains, including multi-tenant telecommunication systems, where the edge-computing mindset is gaining wider adoption to offload some of the computational burden from centralized servers to devices in edge networks. However, ML operations are getting more resource-intensive due to the growing complexity of models and increasing amounts of training data, leading to a rise in environmental sustainability concerns. Hence, further effort is required to measure and decrease the carbon emissions of compute-intensive ML pipelines.

In this work, an existing Graph Neural Network (GNN)-based anomaly detection model, namely GATAKU [6], is deployed in an FL setting, assuming a privacy-critical scenario in which each node in the cluster seeks to preserve the confidentiality of its data. The primary aim is to train a global model while preserving privacy and achieving an acceptable level of accuracy in detecting anomalous network packets. Moreover, the carbon footprint is analyzed and reduced via a smart client selection technique, thereby alleviating the environmental impact of FL pipelines.

The contributions of this study are given below:

- GATAKU [6] is deployed in an FL setup, where an accurate model is trained without sharing sensitive data.
- A Differential Privacy (DP) mechanism is incorporated into the FL pipeline for resisting certain privacy threats.
- An ML-based algorithm is implemented to estimate future carbon emissions of FL clients, which is used to decrease the carbon footprint of the FL process.

ISBN 978-3-903176-82-9

II. Related Work and Background

This section provides a brief literature review of GNN-based anomaly detection and an overview of the FL paradigm.

A. Anomaly Detection using GNNs

Although only a limited number of studies have explored the application of GNN models for anomaly detection in network environments, significant effort has been devoted to this topic. In [7], Naseer et al. explored various deep neural network architectures to enhance anomaly detection techniques. In [8], Siegel merged traditional ML techniques with advanced neural network architectures as a potential technique for anomaly detection. In [9], Kisanga et al. introduce a novel graph-based framework for addressing long-term stealthy threats.

A study by Hwang et al. [10] provides a survey of the current state and challenges in GNN-based anomaly detection, exploring novel architectures, including one-class graph transformation learning [11], hybrid-order graph attention network [12], hop-count-based model [13], etc. Nevertheless, these studies assume a centralized learning environment, which may not be feasible for privacy-sensitive scenarios.

B. Federated Learning

First proposed in 2016 [5], FL has created a new ML paradigm in which each device trains a local model using its own dataset and then shares its model parameters. Commonly used terms in the FL domain are clients and the parameter server (i.e., the aggregator), where the former represent the set of computation devices that join the FL process, and the latter is the unit that performs the aggregation of the local model parameters (i.e., weights) from the clients to construct the global model parameters. An FL process involves multiple iterative rounds, in which every client performs local computations, shares their model weights with the server, after which the server aggregates these parameters and distributes the global model parameters back to clients.

While FL provides a fundamental level of data protection, the privacy of participants may still be violated under a curious aggregator who may obtain knowledge on clients' sensitive data from purely shared model weights [14]. To thwart such threats, many methods are incorporated into FL, including the DP mechanism, which adds a level of noise to model weights before sending them to the aggregator [15].

Carbon awareness of FL processes is also becoming a research direction [16]. While there exist novel studies for measuring and optimizing the carbon footprint of FL pipelines, such as FedCarbon [17], FedGreen [18], FedSynthesis [19], CAFE [20], and FedZero [21], these frameworks usually rely on manual user input or a type of simulation for estimating the carbon footprint. Moreover, studies in this area rarely analyze additional privacy-enhancing methods, such as DP, to achieve a combination of carbon- and privacy-awareness.

To this end, a novel approach is needed to incorporate DP and measurement-based carbon footprint optimization for sustainable and privacy-preserving FL, which could power cybersecurity solutions in the cloud.

III. FL-GATAKU: Design and Implementation

This section describes how GATAKU [6] is modified to operate in an FL environment, focusing on the design and implementation of the proposed approach.

A. Testbed

The leveraged testbed consists of six VMs, each running Ubuntu 22.04. The resource specifications of these nodes are shown in Table I. One VM is reserved for the aggregation server of the FL process, while the other five form the Kubernetes cluster. Among these five VMs, one acts as the control plane, while the rest serve as worker nodes. Note that in this setup, the control plane is not restricted to only managing the nodes, as it also undertakes some workload, allowing the utilization of five VMs in the cluster to run pods and participate in the FL process.

Table I
Cloud Infrastructure Specifications

VM Role	CPU	RAM	SSD
Control-plane / FL Client	8 vCPU	16 GB	160 GB
Worker / FL Client (x4)	4 vCPU	8 GB	80 GB
FL Aggregator	16 vCPU	32 GB	320 GB

B. Data Collection and Model Development

In the Kubernetes cluster, a simple web-based application comprising a front-end service, a back-end service, a database service (e.g., PostgreSQL), and a data service is deployed. Multiple namespaces are used to simulate a multi-tenant environment, where different instances of the same application are isolated from each other.

For threat modeling, GATAKU considers three types of attacks: i) Denial of Service (DoS), ii) Port Scan, and iii) ZAP Scan. While the Hey application tool and Nmap are used to perform the first two attacks, respectively, the ZAP Scan attack is conducted by the OWASP ZAP tool and targets specifically the front-end component of the application by trying to identify vulnerabilities. These attacks are triggered manually while the application runs in the cluster, enabling the system to collect both benign and malicious network data. After a 2-hour run of the application and a set of manually triggered attacks, the collected raw data is preprocessed for normalization and to form a graph structure based on network packet similarity, which is then fed into the GNN model.

For anomaly detection, GATAKU employs a Graph Convolutional Network (GCN) model to identify connections among network packets, enhancing its capability of detecting anomalous traffic. During inference, the test data is also formed into a graph, and the GCN model identifies anomalous packets. For more details on the data collection process and the developed GCN model, the reader is referred to the study [6].

C. Federated Learning Setup

For the FL part of the study, the Flower framework [22] is leveraged, which works agnostically to the underlying library, such as PyTorch, Tensorflow, or scikit-learn.

1) Motivation of FL: The primary motivation for leveraging FL in this setup is that one node in the cluster may be more vulnerable to a specific attack type, while the others are not, thereby encouraging malicious actors to target that node for a more successful attack. In this case, the local dataset of the specific node will contain attack data, which would be absent from other nodes' datasets. Therefore, when an attacker attempts this attack type against other nodes, a model trained solely on the local dataset would be insufficient to distinguish such activity. In this heterogeneous attack data scenario, leveraging FL helps build a more comprehensive model that can detect various types of malicious flows.

2) Carbon Consideration: One straightforward way to reduce carbon emissions in an FL process is to reduce the energy consumption. However, lower energy consumption does not necessarily imply a lower carbon footprint, since the carbon footprint is also affected by the Carbon Intensity (CI) factor, which represents the amount of CO_2 emitted per unit of energy consumed. The CI factor is primarily determined by the characteristics of the power grid that fuels the device.

a) Carbon Emission Tracking: As a means of tracking the carbon emissions of clients during an FL process, CodeCarbon [23] can be used, as it also accounts for the CI factor when estimating the energy consumption resulting from a computational task. In the proposed design, each FL client adopts CodeCarbon to measure its carbon footprint, caused by their local training during each FL round.

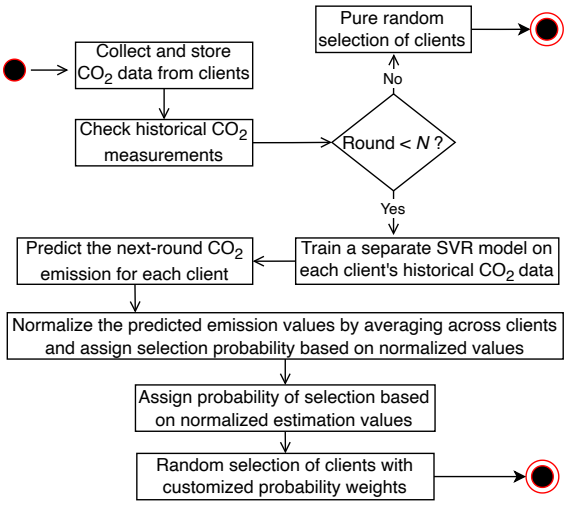


Figure 1. Proposed carbon-aware client-selection algorithm

b) Carbon Emission Estimation: To reduce the carbon footprint of an FL process, it is not sufficient to only track each client's emissions; it is also necessary to estimate the emission values for subsequent rounds. Here, a heuristic approach is adopted, as shown in Figure 1. The main idea is that the server stores each client's CO_2 emissions on a round-by-round basis. For the first N rounds, which is a configurable parameter, the default random client selection is applied due to insufficient data points. After N rounds, it is assumed that sufficient data samples are collected and the next round's CO_2 emissions for each client are predicted by training a Support Vector Regressor (SVR) on historical CO_2 data. Then, these estimates are averaged for normalization, and each client is assigned a probability score based on the next round emission estimate. Then, clients are selected randomly with an adjusted set of probabilities. With this approach, the randomness factor is preserved to ensure fairness across clients while favoring those likely to produce a lower carbon footprint.

D. Overall System Architecture

Based on the previous discussion on the design and implementation, the overall architecture of the proposed system is provided in Figure 2. Each client sends their carbon footprint, tracked by their CodeCarbon instance, along with the local model parameters to the aggregation server each round. The server then stores CO_2 emissions per client on a round-by-round basis and adjusts the client selection mechanism based on the estimated emissions for the next round.

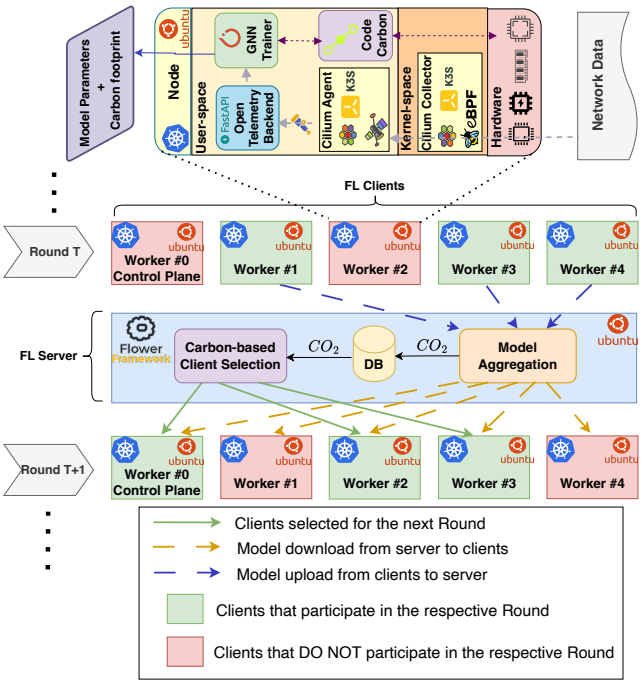


Figure 2. FL-GATAKU architecture

Note that each client adds a certain level of noise, as configured by the DP mechanism, to the computed model weights before uploading them to the aggregator. It is crucial to determine the optimal noise level, as a value that is too high may distort the model and degrade performance, while one that is too low may be insufficient for privacy preservation.

IV. FL-GATAKU: Evaluation

In the experiment phase, the system is evaluated on various scenarios: i) the effectiveness of the FL-based anomaly detection approach compared to training individual per-node models, ii) an investigation of FL-enhanced GATAKU with an incorporated Differential Privacy (DP) scheme, and iii) the capability of the carbon-aware client selection technique that tackles the carbon footprint and detection performance trade-off, in both with DP and without DP cases.

A. FL vs Per-Node Training

In the first set of experiments, the effectiveness of FL-GATAKU is analyzed by deploying GATAKU into each node individually and using its own local dataset for per-node training.

The considered scenario makes the following assumptions:

- Each node collects and stores its own inter-container communication data, which is sensitive. Therefore, local datasets cannot be consolidated into a single dataset.
- It is possible that some nodes have specific weaknesses that render them more vulnerable to certain attacks, resulting in heterogeneous datasets across clients.
- Over time, nodes might develop additional vulnerabilities, making them prone to different types of attacks that were not applicable before.

Given the assumptions outlined in this scenario, each developed ML model is tested against datasets of different nodes to obtain generalizable results, following the approach below:

1) Each local dataset is divided into mutually exclusive training, validation, and test splits in the ratio of 60, 20, and 20, respectively.
2) For the baseline, a separate ML model is trained per node, using its own local training and validation split. For the FL part, a global model is trained using training and validation splits of all nodes in a distributed manner.
3) Each of the six ML models, i.e., five locally trained models plus one globally trained model via FL, is tested against all five local test splits.
4) This process is repeated ten times, and the mean value of metrics across these ten trials is reported.

Table II shows the accuracy values of each case, illustrating that there are not many significant improvements of the FL-based model over per-node models in most cases. However, the cases where the model is trained with the Worker 3 dataset and tested with Worker 1 or Worker 4 achieve an accuracy of approximately 97%, whereas the FL-based model can achieve 99% accuracy with the same test sets, providing at least 2% improvement. Moreover, the standard deviation values are reduced from the order of 10^{-4} to the order of 10^{-5} when switching to the FL approach, showcasing another advantage.

In addition, the extremely low numbers of ZAP scan packets across all datasets make it difficult to assess the model's capability to detect these rare events. As part of a deeper evaluation in this regard, the recall metric for ZAP scan packets is investigated. Table II also shows the recall values for each combination of training and testing datasets, as well as the collaboratively trained FL model, averaged across 10 experiments. Per-node training results in severe variation, from complete failure of detection (i.e., 0%) to a relatively high level of detection (i.e., 73%), whereas the FL model shows much more stable performance. The only case in which a per-node model outperforms the FL is when the model is both trained and tested with Worker 3 datasets. In all other cases, FL proves to be more advantageous than per-node training.

B. FL with DP vs FL without DP

Because of potential privacy threats, such as membership or sample inference [14], baseline FL may be insufficient to prevent data leakage. For this reason, a DP scheme is incorporated into the FL pipeline, and an analysis is conducted on how varying DP parameters affect the performance of the converged GATAKU model.

In these experiments, each FL run is configured to last for 20 rounds, and the DP parameters provided by the Flower framework are utilized: the privacy budget (ϵ), tolerance for privacy guarantee (δ), sensitivity (S), and clipping rate (C). The tolerance, δ, is arbitrarily set to 0.02 for all experiments, indicating that the DP scheme holds for 98% of the time. Each of the other parameters is evaluated in a controlled experiment, by fixing others, and the results are displayed in Figure 3.

It is observed that increasing ϵ has a positive effect on the convergence speed of the model, as shown in Figure 3a. The final model performance, on the other hand, is not severely impacted by varying privacy budget values. It can be seen that the cases $\epsilon = 0.10$ and $\epsilon = 0.05$ provide the same f1-score at round 20, while smaller ϵ values result in a lower f1-score.

Regarding the sensitivity parameter, the results are similar to those in the privacy budget analysis. Figure 3b displays that higher values of S help with the convergence process and a very low value of sensitivity, such as 0.2, causes a drop in the final model performance. Surprisingly, it is also observed that when $S = 0.4$, the final f1-score is as high as the cases $S = 1.0$ or $S = 0.8$, if not slightly higher.

On the other hand, the clipping rate analysis produces more distinctive results, as illustrated in Figure 3c. Lower values, such as 0.3 and 0.5, significantly degrade the initial performance, while affecting convergence time and the final model performance in a negative way, too. While increasing C generally enhances the final model performance, the cases $C = 0.7$, $C = 0.9$, and $C = 1.0$ yield no major difference.

Overall, it is shown that adopting DP slightly reduces the effectiveness of the FL process, an acceptable cost for achieving a higher level of privacy. Also, the findings of this experiment suggest that the following configuration of DP parameters might have minimal effect on the converged model performance: $\epsilon = \mathbf{0.05}$, $\mathbf{S = 0.4}$, $\mathbf{C = 0.7}$, $\delta = \mathbf{0.02}$. Therefore, this configuration is adopted in the following carbon-aware FL experiments with DP scenario.

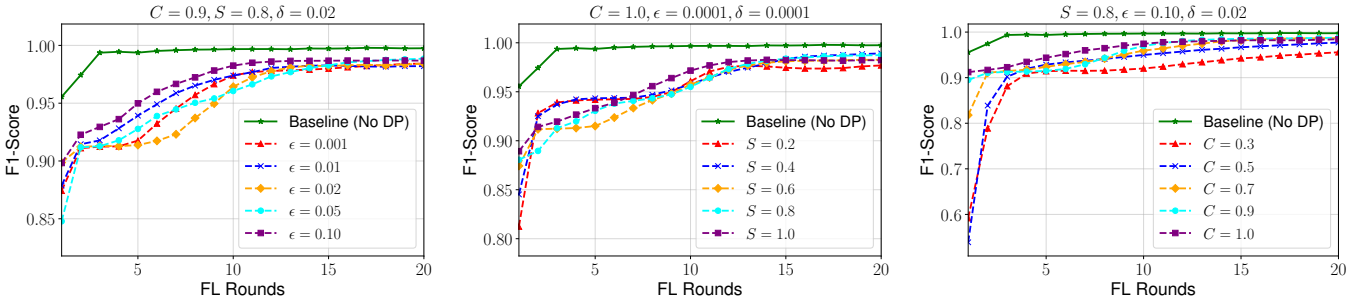
C. Carbon-aware FL vs Standard FL

In this experiment, the aim is to investigate the effects of the proposed carbon-aware FL approach against a regular FL setup. For the baseline, the standard FedAvg method is chosen with completely random client selection. Two different cases are considered: i) where all parties are completely honest, and ii) where parties are honest-but-curious. While a basic FL deployment is sufficient for the former, the latter case requires additional privacy-enhancing methods, such as DP. The parameter N, representing the number of rounds for data collection without carbon footprint estimation, is set to 5 here.

Table II

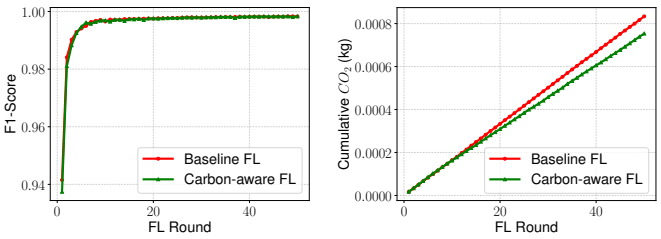
PERFORMANCE METRICS OF THE ANOMALY DETECTION MODELS: W_i INDICATES THE DATASET OF WORKER NODE WITH INDEX i, WHILE **W. ACC.** AND **ZAP REC.** REPRESENT THE WEIGHTED ACCURACY ACROSS CLASSES AND THE RECALL RATE OF ZAP SCAN SAMPLES, RESPECTIVELY

		Testing									
		W_0		W_1		W_2		W_3		W_4	
		W. Acc.	**ZAP Rec.**	**W. Acc.**	**ZAP Rec.**	**W. Acc.**	**ZAP Rec.**	**W. Acc.**	**ZAP Rec.**	**W. Acc.**	**ZAP Rec.**
Training	W_0	0.9989	0.30	0.998	0.83	0.9986	0.59	0.9976	0.26	0.9987	0.39
	W_1	0.9985	0.04	0.9985	0.02	0.9981	0.01	0.9967	0.08	0.9986	0.06
	W_2	0.9982	0	0.9973	0.01	0.9988	0	0.9975	0	0.998	0
	W_3	0.9819	0.57	0.9786	0.42	0.9925	0.26	0.9988	0.73	0.9738	0.47
	W_4	0.9977	0.34	0.9974	0.30	0.9979	0	0.9961	0.26	0.9989	0.06
	FL	0.9985	0.61	0.9979	0.56	0.9984	0.38	0.9977	0.67	0.9984	0.70



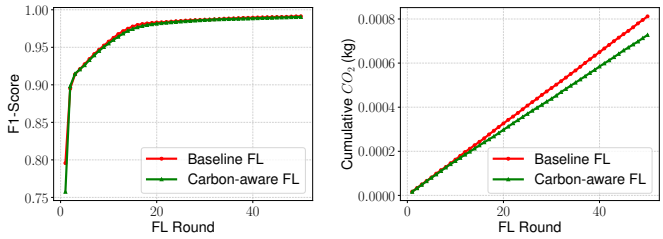
(a) The effect of varying privacy budget: ϵ (b) The effect of varying sensitivity: S (c) The effect of varying clipping rate: C

Figure 3. The effects of differential privacy parameters in FL-GATAKU



(a) F1-score comparison (b) Cumulative CO_2 comparison

Figure 4. Carbon-aware FL against standard FL, without DP



(a) F1-score comparison (b) Cumulative CO_2 comparison

Figure 5. Carbon-aware FL against standard FL, with DP

Table III

PER-ROUND CO_2 AND FINAL-ROUND F1-SCORE IN CASE 1

	$CO_2(10^{-5}kg)$		F1-Score	
Methods	μ	σ	μ	σ
Baseline (FedAvg)	1.667	0.060	0.9983	3.7×10^{-4}
Carbon-aware FL	**1.507**	0.098	0.9983	3.4×10^{-4}

1) Case 1 - Without DP: Due to the assumption that all parties are honest, a basic FL setup is considered sufficient here. For experiments, 20 trials are conducted for both the proposed carbon-aware FL and the baseline FL methods, each consisting of a fixed number of 50 rounds. In each round, three out of five clients are selected to participate in the training process. Evaluations are conducted for both the global model performance and the overall carbon emission, as given in Figure 4. At a first glance, Figure 4a shows that both the baseline model and the proposed model converge rather fast, while Figure 4b clearly illustrates how the carbon-aware FL manages to reduce the cumulative carbon footprint.

a) Carbon Emission: The average carbon emission per round is given in Table III. It is observed that the proposed approach achieved approximately 9.6% reduction in carbon footprint, with a slightly higher standard deviation across 20 trials. However, to ensure this difference is meaningful, ANOVA (Analysis of Variance) test was applied to the carbon footprint values, and a p-value of 3.58×10^{-7}, identifying a significant difference in the two approaches.

b) Global Model Performance: The impact of incorporating carbon-aware client selection on the model performance is also examined in Table III, where the mean f1-scores at round 50 are reported. Given the same mean across trials and a very minor difference in standard deviation, these findings suggest that the carbon-aware approach has no impact on GATAKU's final model performance. Plus, ANOVA test on the final model performance yielded a p-value of 0.986, indicating no statistically significant difference.

2) Case 2 - With DP: In this one, it is assumed that the parties, especially the aggregator, are honest-but-curious, thereby constraining further privacy-enhancing methods for the FL process. The configuration of Case 1 is exactly repeated, albeit with an additional DP scheme where the parameter values are

selected from the results reported in Section IV-B. Results are displayed in Figure 5, with f1-score and cumulative carbon emissions are shown in Figure 5a and Figure 5b, respectively. Although both the baseline and the carbon-aware FL show similar model performance at the end, the convergence process exhibits a serious delay under a DP scheme, compared to Case 1 without DP. This is quite expected, as the noise added by DP negatively impacts model training. On the other hand, the carbon footprint cumulation is almost the same as Case 1, where the proposed approach shows a visible improvement.

Table IV
PER-ROUND CO_2 AND FINAL-ROUND F1-SCORE IN CASE 2

	$CO_2(10^{-5}kg)$		F1-Score	
Methods	μ	σ	μ	σ
Baseline (FedAvg)	1.624	0.070	0.9913	2.8×10^{-3}
Carbon-aware FL	**1.455**	0.086	0.9904	1.9×10^{-3}

a) Carbon Emission: Table IV provides the average carbon emission per round. Once again, the proposed approach reduces the carbon footprint severely, by approximately 10.4%. The significance of the difference is also supported by the ANOVA test applied, which yielded a p-value of 1.3×10^{-5}. It is reasonable that the results are almost the same as those in Case 1 without DP, as the computational cost of DP is extremely low.

b) Global Model Performance: The f1-score values across FL rounds for both methods are also given in Table IV, where the baseline slightly exceeds the carbon-aware FL by 0.09%. Applying ANOVA test resulted in a p-value of 0.27, which indicates that this difference was not significant.

V. CONCLUSION AND FUTURE WORK

This paper presented an application of Federated Learning (FL) to a GNN-based anomaly detection system, namely GATAKU, while promoting a carbon-reduced and privacy-preserving FL pipeline via a smart client selection mechanism and a DP scheme. Through various experiments, the study revealed that the FL-powered GATAKU has a clear advantage over per-node training in a heterogeneous client setup. Moreover, the proposed carbon-reduced approach achieved significant carbon footprint reduction while preserving final model performance in both DP and non-DP cases. These results illuminate a potential path of carbon-reduced FL processes that could power up cybersecurity and privacy applications.

Future work plans include incorporating additional privacy-enhancing techniques, such as multi-party computation, and analyzing their impacts on carbon-aware FL processes. Also, a fairness analysis of the client selection mechanism should be conducted to ensure there is no bias towards certain clients.

ACKNOWLEDGMENT

This work was supported partially by (a) the University of Zürich UZH, Switzerland and (b) the Horizon Europe Framework Program's project NATWORK, Grant Agreement No. 101139285 funded by the Swiss State Secretariat for Education, Research, and Innovation SERI, under Contract No. 22.00642.

REFERENCES

[1] D. Merkel, "Docker: lightweight Linux containers for consistent development and deployment," *Linux J.*, vol. 2014, Mar. 2014.
[2] E. Zeydan, S. Arslan, and Y. Turk, "A cloud native journey for telecommunication networks: Components, applications and open challenges," *ACM Comput. Surv.*, vol. 58, Apr. 2026.
[3] M. Ali, A. Arunasalam, and H. Farrukh, "Understanding users' security and privacy concerns and attitudes towards conversational AI platforms," in *2025 IEEE Symposium on Security and Privacy (SP)*, 2025.
[4] A. Kanso, S. Oks, M. Elzeiny, and G. Virdi, "Towards enabling hostile multi-tenancy in Kubernetes," in *Proceedings of the SC '25 Workshops of the Int. Conf. High Perform. Comput. Netw. Storage Anal.*, SC Workshops '25, (New York, NY, USA), p. 172–178, ACM, 2025.
[5] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Agüera y Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," *arXiv e-prints*, p. arXiv:1602.05629, Feb. 2016.
[6] M. Osswald, T. Schönenberger, G. Cantali, W. Soussi, and G. Gür, "Anomaly detection in microservices architecture using graph neural networks," in *2025 33rd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP)*, IEEE, 2025.
[7] S. Naseer *et al.*, "Enhanced Network Anomaly Detection Based on Deep Neural Networks," *IEEE Access*, vol. 6, pp. 48231–48246, 2018.
[8] B. Siegel, "Industrial Anomaly Detection: A Comparison of Unsupervised Neural Network Architectures," *IEEE Sensors Letters*, vol. 4, pp. 1–4, Aug. 2020.
[9] P. Kisanga, I. Woungang, I. Traore, and G. H. S. Carvalho, "Network anomaly detection using a graph neural network," in *2023 International Conference on Computing, Networking and Communications (ICNC)*, pp. 61–65, 2023.
[10] H. Kim, B. S. Lee, W.-Y. Shin, and S. Lim, "Graph anomaly detection with graph neural networks: Current status and challenges," *IEEE Access*, vol. 10, pp. 111820–111829, 2022.
[11] C. Qiu, M. Kloft, S. Mandt, and M. Rudolph, "Raising the Bar in Graph-level Anomaly Detection," *arXiv e-prints*, p. arXiv:2205.13845, 2022.
[12] L. Huang *et al.*, "Hybrid-order anomaly detection on attributed networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 12, pp. 12249–12263, 2023.
[13] T. Huang, Y. Pei, V. Menkovski, and M. Pechenizkiy, "Hop-count based self-supervised anomaly detection on attributed networks," in *Machine Learning and Knowledge Discovery in Databases*, (Cham), pp. 225–241, Springer International Publishing, 2023.
[14] J. Chen *et al.*, "When federated learning meets privacy-preserving computation," *ACM Computing Surveys*, vol. 56, Oct. 2024.
[15] K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, T. Q. S. Quek, and H. Vincent Poor, "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3454–3469, 2020.
[16] D. Thakur, A. Guzzo, G. Fortino, and F. Piccialli, "Green federated learning: A new era of green aware AI," *ACM Computing Surveys*, vol. 57, Mar. 2025.
[17] Y. Li, T. Ouyang, X. Chen, and X. Cao, "FedCarbon: Carbon-efficient federated learning with double flexible controls for green edge AI," in *2024 IEEE/ACM 32nd International Symposium on Quality of Service (IWQoS)*, 2024.
[18] A. Abbasi *et al.*, "FedGreen: Carbon-aware federated learning with model size adaptation," in *2024 IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 1352–1358, 2024.
[19] G. Cantali, G. Gür, and B. Stiller, "FedSynthesis: A flower-based framework for carbon-reduced federated learning," in *Proceedings of the 18th IEEE/ACM International Conference on Utility and Cloud Computing*, UCC '25, (New York, NY, USA), ACM, 2025.
[20] J. Bian, L. Wang, S. Ren, and J. Xu, "CAFE: Carbon-aware federated learning in geographically distributed data centers," in *Proceedings of the 15th ACM International Conference on Future and Sustainable Energy Systems*, e-Energy '24, (New York, NY, USA), p. 347–360, ACM, 2024.
[21] P. Wiesner *et al.*, "FedZero: Leveraging renewable excess energy in eederated learning," in *Proceedings of the 15th ACM International Conference on Future and Sustainable Energy Systems*, e-Energy '24, (New York, NY, USA), p. 373–385, ACM, 2024.
[22] D. J. Beutel *et al.*, "Flower: A Friendly Federated Learning Research Framework," *arXiv e-prints*, p. arXiv:2007.14390, July 2020.
[23] "Codecarbon." https://codecarbon.io/, 01 2025. [Accessed: 04.05.2026].