Demo: Highlighting the Limits of Federated Learning in Intrusion Detection

Léo Lavaur
IMT Atlantique, IRISA, CyberCNI
leo.lavaur@imt-atlantique.fr

Yann Busnel
IMT Nord Europe, IRISA
yann.busnel@imt-nord-europe.fr

Fabien Autrel
IMT Atlantique, IRISA
fabien.autrel@imt-atlantique.fr

***Abstract*—Federated learning (FL) is a distributed learning paradigm enabling participants to collaboratively train a machine learning (ML) model. In security-oriented tasks, FL can be used to share attack knowledge, without sharing participants' local data. Recent research results reveal that highly heterogeneous data distributions can prevent federations from converging towards an appropriate global model. Moreover, maintaining trustworthiness is challenging, as FL-based collaborative intrusion detection systems (CIDSs) are vulnerable to malicious updates.**

In this demonstration paper, we present critical examples of these challenges using a set of standardized public datasets and a dedicated automation tool. We review the impact of heterogeneity using different data-distribution, before looking at a scenario with malicious actors.

***Index Terms*—demonstration, federated learning, intrusion detection, heterogeneity, adversarial mitigation, data poisoning.**

I. Introduction

Due to the increasing importance of privacy-related constraints, organizations are reluctant to share their data with third parties. This is especially true in the context of intrusion detection systems (IDSs), where the data is highly sensitive. To address this issue, federated learning (FL) has been proposed as a solution to enable collaborative intrusion detection systems (CIDSs). FL is a distributed learning paradigm where participants share trained models instead of raw data [1].

However, FL suffers from several limitations, especially in terms of participants' heterogeneity. Indeed, the default aggregation algorithm, `FedAvg` [1], requires that the data distributions are similar enough to converge to a suitable global model. This is not always the case in practice, and the performance of the algorithm decreases when the data distributions are too different. On the other hand, if the data distributions are too similar, the benefits of FL are negligible.

In this paper, we present a demonstrator to highlight the limits of these federated intrusion detection systems (FIDSs) using FL. Because the demonstrator is based on a purposely built evaluation framework, the parameters of the experiments can be easily adjusted by the attendees, making the demonstration highly interactive. Section II summarizes the state of the art on FL in intrusion detection. Section III presents the demonstration setup and a set of illustrative scenarios.

This research is part of the chair CyberCNI.fr with support of the FEDER development fund of the Brittany region.

II. Collaborative Intrusion Detection with Federated Learning

Since its introduction, FL has been applied to FIDSs in numerous works, using supervised or unsupervised learning, or even technics in between [2]. However, a majority of these works focus on the performance of the algorithm, and do not always consider the impact of data partitioning on the aggregated model. Furthermore, few of them share public implementations of their solution allowing to quickly illustrate limits of FL in IDS contexts.

A. Federated Learning on non-IID data

In the FL foundation paper [1], the authors emphasize on non independent or identically distributed (NIID) data being one of the key attributes of FL, alongside the unbalanced overall distribution. They notably present a *pathological*-NIID situation using MNIST [3], a digit recognition dataset, where each client is given only two digits, *e.g.* 3 and 7. More recent papers consider alternative NIID use cases, deemed more realistic. For instance, Huang *et al.* [4] present a *practical*-NIID use case, where participants can share similarities. This is particularly suited for cross-silo use cases, such as CIDSs.

B. Data Partitioning in IDS contexts

Pathological-NIID partitioning is rarely seen in IDS binary-classification tasks, as they typically require both benign and malicious training data. Therefore, a common NIID partitioning scheme is: 1) *pathological*-NIID of the attack classes, *e.g.* one or two class per client; and 2) independent and identically distributed (IID) benign samples. Campos *et al.* [5] also review other partitioning settings based on the ability to separate data by client IP in public datasets. They also artificially build balanced IID partitions by dropping attack samples until a specific Shannon entropy threshold window is reached for the local distribution. This approach is however more suited for cross-device use cases, as each client receives the data from one device only. Overall, NIID data for a cross-silo network-based intrusion detection system (NIDS) context is typically one of:

(a) distributing a dataset among clients, before removing samples from n attack classes from each client; or
(b) distributing the benign data among clients, before giving samples from n attack classes to each client, with or without class overlap.

TABLE I: Parameters used for all scenarios.

Parameter	Notation	Value
Federated Learning		
Number of rounds	R	10
Local epochs per round	ε	10
Number of clients	C	4
Local Training		
Neurons of the (2) hidden layers		128
Activation function (hidden layers)		ReLU
Activation function (output layer)		Softmax
Batch size	β	512
Learning rate	η	0.001
Datasets		
Number of features	D	39
Number of samples	N	100,000

C. Threats against Federated Learning

Given the distributed nature of FL in a CIDS use case, the quality of the model updates shared by clients can vary, and low-quality updates can negatively impact the performance of the aggregated model. Existing works often focus on model poisoning, as it can produce finer attacks than data poisoning [6]. Indeed, the ability to control the model update provides more flexibility to the attacker, as they can target specific parameters of the model. However, data poisoning attacks are easy to implement and configure, while still yielding impactful results [7]. Furthermore, considering malicious actors also protects against clients uploading low-quality contributions, since they are likely to produce less impactful updates.

III. Exhibiting the Limits of FIDSs

This demonstration spans over four specific scenarios, each highlighting a specific aspect of the considered challenges. The first three (Sections III-B to III-D) target different heterogeneity scenarios, ranging from homogeneous dataset partitioning to completely independent data sources. The last scenario (Section III-E) focuses on poisoning attacks against FL, where malicious participants try to degrade the performance of the global model.

A. Setup

For the sake of reproducibility, these experiments are made using public datasets whose features have been standardized in [8]. The authors provide modified versions of the original datasets with this feature set (based on NetFlow v9) and using nProbe[1]. Namely, we use the following (adapted) datasets:

- UNSW-NB15 [9] is produced using the IXIA PerfectStorm tool on the Cyber Range Lab of UNSW Canberra. The traffic is a hybrid set of real modern normal activities and synthetic contemporary attack behaviors, grouped in 9 attack classes.
- Bot-IoT [10] is another dataset generated at USNW, using a realistic smart home environment setup, completed by IoT devices. It focuses on the detection of IoT botnet attacks, the DoS and DDoS classes being the most represented. This dataset is highly unbalanced, as the majority of the traffic is malicious.
- ToN_IoT [11] is yet another dataset generated by the same team, containing IoT/IIoT telemetry data, network traffic, as well as system logs. The network dataset contains 9 attack classes, including Ransomware, Scanning, and XSS.
- CSE-CIC-IDS2018 [12] is a dataset generated by the Canadian Institute for Cybersecurity in collaboration with the Communications Security Establishment (CSE). The traffic is collected on a large-scale infrastructure deployed on AWS. It contains 14 attack labels, grouped in 6 attack classes.

To generate the different scenarios, we build an evaluation framework for FL called Eiffel[2], which relies on Flower [13], a modular FL framework. Eiffel is a Python library that provides a set of tools to automate the evaluation of FL algorithms, such as instantiating various types of data distribution, local models, and aggregation strategies. It further provides multiple label-flipping attacks, and automates metric collection and plotting to quickly evaluate the impact of each parameter.

To assess the impact of a scenario on the federation, we evaluate the global model on each participant's test set and collect different performance metrics. The results are averaged over the different participants to obtain the global model's performance. We select the F1-score as the main metric for its focus on positive samples, but the same methodology can be applied to other metrics. To assess the performance of a model trained locally, we define a `FedNoAgg` strategy, where local models are kept by participants at the end of each round. Therefore, models are trained during $\varepsilon \times R$ local epochs, where R is the number of rounds and ε is the number of local epochs per round, instructed by the server. Table I summarizes the parameters used for all scenarios.

B. Scenario 1: IID Data

The first scenario is the simplest one, where the data is partitioned in IID settings. Each participant receives $\frac{N}{C}$ samples, after shuffling the dataset. Figure 1 presents the results of this scenario based on the global model's F1-score. There are virtually no differences between the `FedNoAgg` and `FedAvg` strategies, since each participant has enough samples of each class to train a suitable local model. Therefore, there are few benefits to using FL in this scenario.

However, this configuration is often found in the literature to evaluate CIDSs based on FL, such as in [14]. While this experiment illustrates the lack of performance gains on IID data, larger-scale setups configurations might benefit from FL. In fact, selecting only a subset of the available participants could obtain similar results while reducing the local computing costs for participants. This setup is thus more akin to a distributed

[1]Available at: https://www.ntop.org/products/netflow/nprobe/

[2]Available at: https://github.com/phdcybersec/eiffel

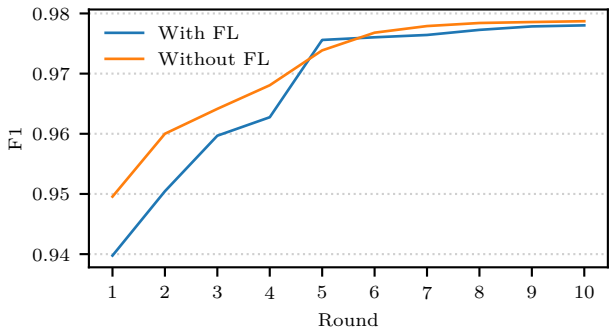

Fig. 1: Global model performance in IID.

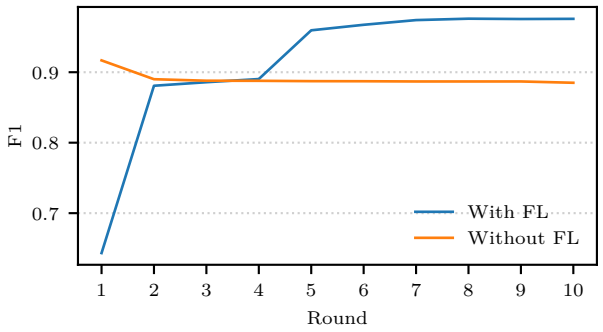

Fig. 2: Global model performance in NIID (same source).

learning approach, where the server is only used to coordinate the training process.

C. Scenario 2: NIID Data from the Same Source

The second scenario highlights the knowledge-sharing capabilities of FL, as it can transfer characteristics of the data distribution between participants. To illustrate this, after partitioning the data as in Section III-B, we randomly drop two classes from each participant's train set. This results in a NIID data distribution among participant, where each one has a different subset of classes. Figure 2 displays the results of this scenario, where `FedAvg` performs significantly better overall than having clients train locally. However, the F1-score hides the fact that some participants can miss entire attack classes in the test set, rather than it being a global model issue.

Specifically, since clients have different subsets of classes, they might be unable to detect some intrusions that are not present in their training data. For example, Table II displays the detection rate (DR) of the first client (`client_0`) in our setup for each attack class, both in local and federated training, along with the number of samples of each class. `client_0` has no samples of the `Infiltration` and `DoS` classes, and therefore cannot detect them, *i.e.* its DR is either 0 or very low. However, the global model is able to detect these classes, as other clients have samples of these classes in their training set. We also see a slight decrease in performance for the other classes (*e.g.*, 99.91 instead of 100 for `DDoS`) due to the aggregation process.

These results indicate that FL can effectively share knowledge between participants, allowing them to detect attacks that are not present in their local training data. This is a key feature of FL in the context of intrusion detection.

D. Scenario 3: NIID Data from Different Sources

While we highlight in Section III-C that FL can benefit from having different datasets per client, to the point where it can share knowledge between participants, the third scenario illustrates the limits of this assumption. CIDS experiments in the literature often evaluate their approach with a scenario close to the ones presented in Sections III-B and III-C, where one dataset is partitioned among participants. However, in practice, participants will likely collect data from different networks, and therefore have different data distributions.

TABLE II: Detection rate (DR) of `client_0` in NIID settings. Rows where knowledge-sharing is visible are highlighted in gray.

Attack class	Samples	DR (local)	DR (federated)
DDoS	176107	100	99.91
DoS	0	2.43	98.57
Bot	1513	100	99.94
Brute force	1299	99.77	99.55
Infiltration	0	0	20.11
Injection	3	100	100

In this third scenario, we test `FedAvg` in this configuration, with each participant having a different dataset. Thanks to the standardized feature set (see Section III-A), we can use the same model architecture for all participants, which is a requirement for `FedAvg`. The class overlap between datasets is also not an issue in this use case, as we focus on binary-classification, which implies that all participants have benign and malicious samples.

The results presented in Figure 3 confirm great performances overall when participants are trained locally. However, the global model's performance is highly impacted by the heterogeneity of the data distributions. This is likely due to the fact that all participants converge to local minima that are too different from each other, and therefore the aggregation do not result in a suitable model for all participants. Other approaches than `FedAvg` have been proposed to address this issue in IDS context, as the one by Popoola *et al.* [15] for instance.

E. Scenario 4: Poisoning Attacks

With the first three scenarios, we have highlighted how the heterogeneity between participants can impact the performance of FL. However, these scenarios assume that participants are honest and respect the protocol. In this last scenario, we demonstrate how FL can be vulnerable to malicious participants, whose goal is to degrade the performance of the global model. To do so, we use poisoning attacks (see Section II), where attackers flip the labels of samples in their training data to degrade the performance of the global model.

Two of the four clients are instructed to perform a label flipping attack on their entire training set. We can observe in local training (Figure 4) that participants identified as

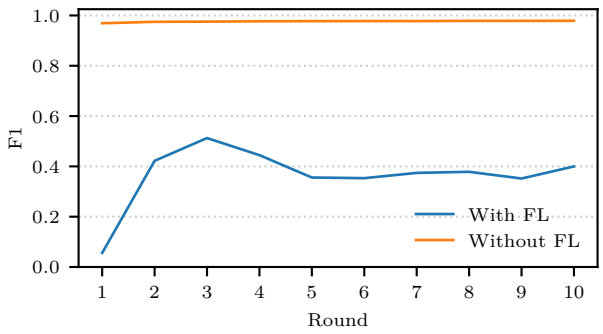

Fig. 3: Global model performance in NIID (different sources).

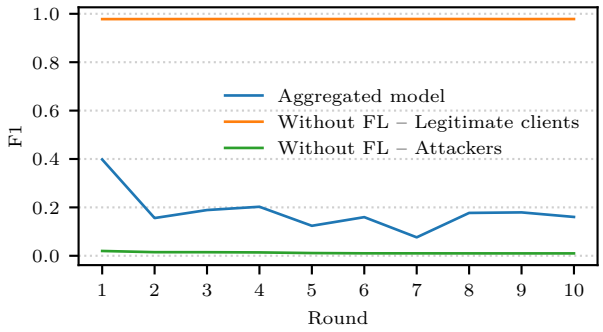

Fig. 4: Global model performance in poisoning attacks.

"Attackers" have a very low DR on their test set, as they literally misclassify all of their testing samples. The two benign participants, on the contrary, reproduce the results of Section III-B, with a high DR on their test set.

In FL however, the global model is impacted by the malicious participants, as illustrated in Figure 4. The participants cannot converge towards a stable global model, as the malicious participants' updates are too different from the others. Due to the miss-classification introduced by the malicious participants, the global model's performance is degraded, and theFF1-score oscillates between 0.1 and 0.2. This is critically low, as it means that the aggregated model either misses a lot of attacks and misclassifies a lot of benign samples.

IV. Conclusion

In this paper, we present a demonstrator that aims at highlighting the limits of FL in the context of intrusion detection. With the help of our dedicated evaluation framework, we show that FL can be highly impacted by the heterogeneity of the data distributions between participants. Furthermore, this demonstration presents a critical scenario of poisoning attacks against FL, where the performance of the algorithm is highly impacted. Extended analyses on this scenario can be found in [16]. This emphasizes on the necessity of counter-measures, in particular works on detecting malicious participants' contributions.

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, 2017.

[2] L. Lavaur, M.-O. Pahl, Y. Busnel, and F. Autrel, The Evolution of Federated Learning-based Intrusion Detection and Mitigation: A Survey. *IEEE Transactions on Network and Service Management*, 2022.

[3] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

[4] Y. Huang, L. Chu, Z. Zhou, L. Wang, J. Liu, J. Pei, and Y. Zhang, Personalized Cross-Silo Federated Learning on Non-IID Data. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

[5] E. M. Campos, P. F. Saura, A. González-Vidal, J. L. Hernández-Ramos, J. B. Bernabé, G. Baldini, and A. Skarmeta, Evaluating Federated Learning for intrusion detection in Internet of Things: Review and challenges. *Computer Networks*, 2022.

[6] M. Fang, X. Cao, J. Jia, and N. Gong, Local Model Poisoning Attacks to Byzantine-Robust Federated Learning. In *Proceedings of the 29th USENIX Security Symposium*, 2020.

[7] Y. Zhang, Y. Zhang, Z. Zhang, H. Bai, T. Zhong, and M. Song, Evaluation of data poisoning attacks on federated learning-based network intrusion detection system. In *Proceedings of HPCC/DSS/SmartCity/DependSys*, 2022.

[8] M. Sarhan, S. Layeghy, and M. Portmann, Towards a Standard Feature Set for Network Intrusion Detection System Datasets. *Mobile Networks and Applications*, 2022.

[9] N. Moustafa and J. Slay, UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In *Proceedings of the 2015 Military Communications and Information Systems Conference (MilCIS)*, 2015.

[10] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset. *Future Generation Computer Systems*, 2019.

[11] N. Moustafa, M. Keshky, E. Debiez, and H. Janicke, Federated TON_IoT Windows Datasets for Evaluating AI-Based Security Applications. In *Proceedings of the 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, 2020.

[12] I. Sharafaldin, A. Habibi Lashkari, and A. A. Ghorbani, Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. In *Proceedings of the 4th International Conference on Information Systems Security and Privacy*, 2018.

[13] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, T. Parcollet, and N. D. Lane, Flower: A friendly federated learning research framework. 2020. arXiv: 2007.14390.

[14] O. Aouedi, K. Piamrat, G. Muller, and K. Singh, Intrusion detection for Softwarized Networks with Semi-supervised Federated Learning. In *Proceedings of the IEEE International Conference on Communications (ICC 2022)*, 2022.

[15] S. I. Popoola, G. Gui, B. Adebisi, M. Hammoudeh, and H. Gacanin, Federated Deep Learning for Collaborative Intrusion Detection in Heterogeneous Networks. In *Proceedings of the 2021 IEEE 94th Vehicular Technology Conference*, 2021.

[16] L. Lavaur, Y. Busnel, and F. Autrel, Systematic Analysis of Label-flipping Attacks against Federated Learning in Collaborative Intrusion Detection Systems. In *Proceedings of the 19th International Conference on Availability, Reliability and Security (ARES)*, 2024.