Tutorial: Federated Learning × Security for Network Monitoring

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

Léo Lavaur
IMT Atlantique / IRISA
Rennes, France
leo.lavaur@imt-atlantique.fr

***Abstract*—Federated Learning (FL) is a Machine Learning paradigm that enables training models across distributed clients without accessing their data. In the context of network security, FL can be used to collaboratively train Intrusion Detection System (IDS) models across multiple organizations, virtually extending the local dataset of each participant. Among the new challenges raised by this approach, the heterogeneity of the clients' environments induce consequent differences in the data distributions, and therefore contributions. Further, identifying and mitigating malicious contributions is made more complex in heterogeneous environments.**

This tutorial introduces the audience to the principles of FL and its application to network security, and more specifically to build Collaborative Intrusion Detection Systems (CIDSs) using FL. We address open challenges on that regard, before focusing on the problem of training on heterogeneous data. Finally, we discuss the issues raised by using FL in the context of network security, with a particular focus on poisoning attacks.

***Index Terms*—Federated Learning, Network Security, Collaborative Intrusion Detection System, Network Monitoring, Data poisoning.**

I. Introduction

The emergence of Federated Learning (FL) has opened new perspectives for collaborative learning across distributed clients. The topic is particularly relevant for the distributed system community, as it echoes to a lot of the challenges faced in this field.

FL has emerged as a promising paradigm for collaborative machine learning, enabling model training across decentralized networks while preserving data privacy. This article delves into the fundamentals of FL, exploring its core principles and applications (Section II). The aim is to establish a solid foundation for understanding the subsequent discussions on FL's applications in collaborative network security and the associated security challenges.

a) Federated Learning for Collaborative Network Security: Section III delves into the application of FL in the realm of network security, specifically on training Collaborative Intrusion Detection System (CIDS) models. Addressing the challenges posed by the heterogeneity of clients' data, the discussion explores strategies to ensure effective model training in diverse network settings.

b) Security Challenges in Federated Learning: The last part (Section IV) of this article navigates through the security challenges inherent in deploying and operating Federated Intrusion Detection Systems (FIDSs). With an in-depth understanding of the dynamics of federations, this part browses vulnerabilities and potential attacks that can compromise FL systems. Specifically, the focus is on poisoning attacks, wherein malicious participants attempt to subvert the global model's integrity.

c) Hands-on: Through examples and hands-on exercises provided in the companion repository [1], the reader should gain insights into practical implementation of FL using Flower [2], an open-source Python framework. Hands-on activities involve constructing a basic CIDS model using Flower and experimenting with real-world network traffic datasets, providing participants with practical insights into tackling the complexities of collaborative network security using FL. Through interactive exercises, readers could also simulate and analyzes poisoning attacks on CIDS models, alongside devising and testing mitigation strategies to safeguard against such threats.

II. Fundamentals of Federated Learning

FL emerges at the intersection of collaborative computing and machine learning paradigms, offering a revolutionary approach to training models across a distributed network of devices without centralized data aggregation. Rooted in the concept of crowdsourcing, where large groups contribute or produce goods and services, FL extends this notion to machine learning, enabling diverse participants, from smartphones to IoT devices, to collaboratively improve model performance while preserving data privacy.

The genesis of FL can be traced back to crowdsourcing platforms like Waze, where users collectively contribute real-time traffic data, or collaborative journalism initiatives, illustrating the power of decentralized contributions. Industrializing crowdsourcing further, FL finds applications in various domains, marking a shift towards harnessing collective intelligence for data-driven tasks.

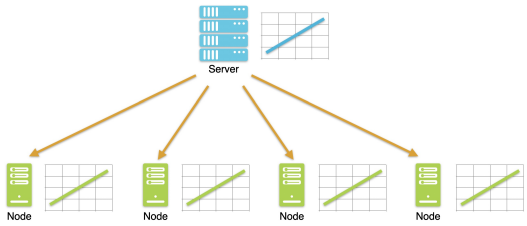

Figure 1. First distribution of the model, from the server to the participating nodes

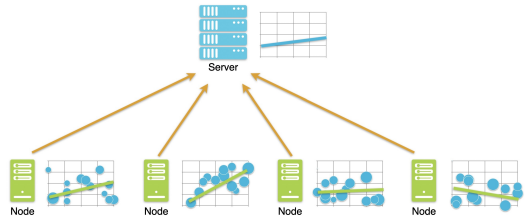

Figure 2. Models modified by nodes, by integrating local data, are returned to the server

In its foundation, FL addresses the limitations of centralized machine learning by distributing the learning process across multiple nodes, each possessing local data and processing capabilities. This distributed approach not only enhances scalability to handle large datasets, but also mitigates privacy concerns associated with sharing sensitive data. Thus, the motivation for FL includes performance improvement with more data, meaningful combination of models and local training at node scale (and not only prediction at the edge). By allowing nodes to collaborate on model updates without sharing raw data, FL ensures privacy compliance with regulations such as HIPAA and GDPR, crucial in sensitive domains like healthcare and advertising.

A. FL in a Nutshell

At its core, unlike traditional centralized learning approaches, FL does not require raw data to be uploaded to a central server. The latter transmits to the participating nodes an initial model, usually generated randomly (*cf.* Figure 1). Then, model updates are computed locally on each device based on its data and then shared with a central server or aggregator (*cf.* Figure 2). These updates are aggregated to construct a global model, which is subsequently refined and redistributed back to the participating devices. This iterative process continues, with each round of communication and aggregation improving the global model's accuracy without compromising the privacy of individual data.

Despite its promising potential, FL faces several challenges, including power consumption, dropped connections or high-latency due to stragglers. Moreover, other issues must be considered, like communication overhead and privacy concerns. These challenges necessitate robust solutions like end-to-end encryption and secure aggregation to safeguard data integrity and confidentiality.

B. Different Approaches of FL

Two variations of FL exists, which are tailored to different contexts and requirements: Cross-Device Federated Learning (CD-FL) and Cross-Silo Federated Learning (CS-FL).

Cross-device FL: In *cross-device* settings, the participating devices are typically heterogeneous and widely distributed, encompassing a massive number of parties, such as smartphones, IoT devices, and personal computers, potentially ranging from thousands to billions, with each device possessing a small dataset. Due to the diversity of devices and their varying computational capabilities, cross-device FL often encounters challenges related to limited availability, reliability, and communication overhead, but offers scalability and adaptability, making it suitable for scenarios where a large and diverse set of devices collaborate on model training tasks.

Cross-silo FL: In contrast, *cross-silo* FL operates within organizational boundaries or distinct data silos, where each silo represents a separate entity or institution. Silos could correspond to different departments within a company, independent organizations, or even geographical regions. Unlike CD-FL, which involves heterogeneous devices, CS-FL typically implies organization with more homogeneous capabilities and more data to train on. Parties in cross-silo FL are more likely to be reliable and consistently available for participation, as they are usually institutional entities with dedicated infrastructure and resources. Yet, entities involved in CS-FL also tend to have considerably greater discrepancies in terms of objectives and data-distributions, and sometimes even model architectures. Cross-silo FL offers more control over data governance and security, as data sharing occurs within predefined organizational boundaries, facilitating compliance with regulatory requirements and privacy policies.

On the other side, two paradigms of distribution exist in the federated approach, each offering unique advantages and challenges, catering to different use cases and requirements.

Server-orchestrated FL: A central server coordinates the training process by managing communication between participating devices and aggregating model updates. The central server plays a pivotal role in distributing model parameters, orchestrating training rounds, and aggregating updates from individual devices. This approach requires global coordination and synchronization, as all communication and aggregation activities are orchestrated by the server. While server-orchestrated FL offers centralized control and streamlined management, it also introduces potential

single points of failure and scalability limitations due to the server's central role.

Fully decentralized FL: There is no central server orchestrating the training process. Instead, devices communicate directly with each other and perform local model updates and aggregations independently. Each device acts autonomously, making decisions regarding model training, aggregation, and synchronization without reliance on a central authority. This decentralized approach eliminates single points of failure and allows for greater scalability, as communication and computation can be distributed across a large number of devices. However, fully decentralized FL may face challenges related to coordination, consistency, and synchronization, especially in scenarios with a vast number of participating devices.

C. Wide Acceptance of the Paradigm

As evidenced by its exponential growth in research (from a few dozens in the first years to thousands of publications today) and real-world deployments, FL stands as a burgeoning field with profound implications for various industries. With open-source libraries like PySyft and TensorFlow Federated facilitating its adoption, FL fosters interdisciplinary collaboration, bridging machine learning, privacy, and networked systems to shape the future of decentralized intelligence.

III. Federated Learning for Collaborative Network Security

Artificial Intelligence (AI) can intervene at various stages of the network security lifecycle, from threat detection to alert correlation and triage. Deep Learning (DL) techniques, in particular, have shown promising results in enhancing the performance of Intrusion Detection Systems (IDSs) by allowing for learning more complex patterns and behaviors, and generalizing to zero- or one-day attacks. Yet, training DL-based IDSs requires large amounts of labeled data to properly learn the underlying patterns of normal and malicious behaviors. In practice, organizations often face challenges in collecting and sharing such data due to privacy concerns, such as sensitive information leakage or regulatory compliance. FL offers a compelling solution to this problem by enabling organizations to collaboratively train IDS models without sharing raw data.

Typical FL applications often imply cross-device settings, with the hypothesis of a single actor trying to learn from multiple devices without accessing their data. The CIDS use case is slightly different, as it usually involves multiple organizations, each owning independent datasets and infrastructures. We refer to CIDS leveraging FL as FIDS, as they are federated across organizations. This context raises new challenges, notably the heterogeneity of the data distributions across organizations. These differences can further vary in terms of monitored traffic (*e.g.*, services, protocols, user behavior), the deployed security solutions, or even the DL models used. The last point is particularly critical, as it prevents the direct aggregation of models as done by `FedAvg` [3].

Simplifying the problem, we can consider the case of a shared model architecture, ensuring the applicability of most FL strategies. Yet, the data heterogeneity remains a significant challenge, as model aggregation of highly heterogeneous data is already identified as an open challenge in FL literature [4].

Previous works [5] provided a comprehensive overview of the state-of-the-art of FIDSs, identifying clear research directions. These challenges span over three main axes:

(i) *Transferability, adaptability, and scalability.* How to deal with a high number of clients and constrained environments? How to learn from heterogeneous data, or heterogeneous clients? How to balance generalization and specialization for model aggregation?

(ii) *Security, trust, and resilience.* How to resist to poisoning and inference attacks against the aggregated models? How to deal with untrusted or malicious participants? How to leverage FL to react to attacks?

(iii) *Local algorithm and aggregation performance.* What is the impact of the hyper- and meta-parameters? How to model behaviors to better characterize traffic? How to improve the raw performance of models?

IV. Security Challenges in Federated Learning

The distributed nature of FIDS opens the way to various attack vectors, ranging from poisoning attacks to privacy breaches. The former aim at altering the global model's behavior, while the latter target the confidentiality of the data used by the participants' in training. We especially focus on the poisoning attacks, as they are particularly relevant in the context of FIDS [6]. Indeed, the ability to manipulate the global model is a significant threat, as it can lead to an overall decrease in the protected system's security. Rodríguez-Barroso *et al.* [7] summarizes the different types of poisoning attacks in a fourfold taxonomy:

(i) *Attack Moment*: whether the attack is performed during the training or the inference phase.

(ii) *Attackers' Objective*: Targeted/Backdoor attacks aim at specific samples or classes, modifying the model's behavior when subjected to specific behaviors, while untargeted poisoning alters the global model uniformly.

(iii) *Poisoned components*: depending on if the attacks alters the training data or the model directly, the impact will vary.

(iv) *Frequency*: one-shot attacks or adaptive/iterative ones. In the latter case, different strategies can be adopted, *e.g.*, increasing the percentage data over time to slowly divert the global model of its original local optimum.

A. Threat modeling in FL settings

Depending on the attacker's positioning and knowledge, different threat models can be considered.

Outsider vs. *Insider:* An outsider has no knowledge of the model or the data used and cannot interact with the system, while an insider has access to the model and its own local data. Further, insiders can impact the model's behavior, and therefore the other participants, while outsiders are limited to listening to the communication channel.

Lone vs. *Colluding Attackers:* Especially in the context of insiders, attackers can act alone or in collusion to maximize their impact. This collusion can be caused by a single entity controlling clients (Sybils) or multiple entities sharing a common interest.

Honest-but-curious vs. *malicious clients:* Honest-but-curious clients follow the protocol but try to learn from the model or the data, while malicious clients actively try to disrupt the system.

B. Mitigation strategies

Fortunately, the community also proposed numerous mitigation strategies against adversarial attacks targeting FL systems, and FIDS by extension. Specifically, multiple works have focused on the development of robust aggregation algorithms and contributions filtering mechanisms, which can mitigate the effect of poisoning attacks. These strategies can be classified into three main categories:

Server-side evaluation: the server evaluates the received contributions on a purpose-built representative dataset. This is mostly inapplicable in Non Identically or Independently Distributed (NIID) settings, as building a representative dataset would imply having access to the clients' data-distribution.

Server-side model comparison: the server compares the received contributions to a reference model, or to each other. In the former, just as in the previous case, the absence of a single-source-of-truth in NIID settings makes this approach difficult. By comparing models to each other, however, the server can detect discrepancies and identify potential malicious contributions. This is the strategy leveraged by `FoolsGold` [8] or `FLAME` [9], although with different approaches and objectives.

Client-side evaluation: the clients are tasked to evaluate other participants' contributions and generate metrics using their local dataset. This removes the need for a single source of truth, while the metrics act as feedbacks that can be used to feed reputation systems.

V. Conclusion

In conclusion, this article has covered the fundamentals of Federated Learning (FL) and its applications in collaborative network security. Participants gained insights into FL principles, practical implementation (using the Flower framework), and its role in training Collaborative Intrusion Detection System (CIDS). FL offers a powerful solution for decentralized data training while preserving privacy. However, deploying Federated Intrusion Detection System (FIDS) poses unique security challenges, such as poisoning attacks. By providing knowledge and tools to tackle these challenges, this tutorial aims to empower researchers and practitioners in utilizing FL for collaborative network security. Moving forward, further research and development efforts are essential to advance FL techniques and enhance the resilience of FL systems against emerging threats in distributed computing environments.

References

[1] L. Lavaur and Y. Busnel, *Hands-on: Fl × security for network monitoring*, https://github.com/phdcybersec/icdcs_2024 – IMT Atlantique / IMT Nord Europe, 2024.

[2] 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.

[3] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *20th intl conf. AIStat*, 2017.

[4] H. Zhu, J. Xu, S. Liu, and Y. Jin, "Federated learning on non-IID data: A survey," *Neurocomputing*, vol. 465, 2021.

[5] 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.

[6] L. Lavaur, Y. Busnel, and F. Autrel, "Demo: Highlighting the limits of federated learning in intrusion detection," in *44th IEEE ICDCS*, 2024.

[7] N. Rodríguez-Barroso, D. Jiménez-López, M. V. Luzón, F. Herrera, and E. Martínez-Cámara, "Survey on federated learning threats: Concepts, taxonomy on attacks and defences, experimental study and challenges," *Information Fusion*, vol. 90, 2023.

[8] C. Fung, C. J. M. Yoon, and I. Beschastnikh, "The limitations of federated learning in sybil settings," in *23rd intl symp. RAID*, 2020.

[9] T. D. Nguyen, P. Rieger, H. Chen, *et al.*, "FLAME: Taming Backdoors in Federated Learning," in *31st USENIX Security*, 2022.