First Steps Towards Federated Learning Network Traffic Detection

Antonio Boiano, Valeria Detomas, Alessandro E. C. Redondi, Matteo Cesana
DEIB, Politecnico di Milano
Milan, Italy
{name.surname}@polimi.it

***Abstract*—Federated Learning (FL) has emerged as a promising machine learning approach which enables cooperative training while preserving data privacy. The FL process involves clients and a parameter server to exchange model parameters over the network, generating unique traffic patterns that can be detected. The identification of FL traffic may be used in diverse applications, ranging from network traffic management to security and privacy enhancement in FL environments. For this purpose, this paper investigates the problem of FL traffic identification. We created a dataset of FL network traces, encompassing different FL training setups. We then developed two classifiers based on Random Forest and Neural Network algorithms and compared their performance, obtaining promising results.**

***Index Terms*—Federated Learning, Network Traffic Classifier, Random Forest, Neural Network.**

I. Introduction

Federated Learning (FL) is a decentralized machine learning paradigm where model training occurs across multiple edge nodes housing local data samples, without the need to exchange the actual data themselves [1]. This setup ensures that data ownership and privacy remain intact as only model updates are shared. In contrast to conventional centralized learning methods, which involve transferring all data to a central server for training, FL boasts several advantages, including enhanced privacy, improved communication efficiency, scalability, and, notably, robustness [2].

Within this context, the TRUSTroke project [3] introduces an innovative AI-based platform to support clinicians, patients, and caregivers in managing acute and chronic phases of ischemic stroke. Central to the project's objectives is the establishment of a FL infrastructure allowing multiple hospitals and clinical sites to collaborate on training AI models for stroke-related predictions without compromising data privacy. This infrastructure prioritizes trustworthiness, robustness, and privacy preservation, aligning with the stringent requirements set forth by the EU Artificial Intelligence (AI) Act. The FL process can be summarized as it follows: (i) a centralized entity, known as the Parameter Server (PS) disseminates a model structure (generally a Neural Network (NN)) with randomly initialized weights to the participating clients, (ii) each client performs a training step using the locally available data and transmits the updated model parameters back to the PS, (iii) the PS aggregates the parameters from the different clients using speficic algorithms (e.g., FedAvg [4]) and disseminates the computed global model back to the clients. Such steps are repeated over a certain number of training rounds until convergence is reached.

978-3-903176-64-5 ©2024 IFIP

Figure 1 shows the traffic pattern (in packets/sec) observed at a client during the FL process. As one can see, such traffic exhibits unique characteristics, such as periodic behaviour (due to the training rounds) and consistent bidirectional data exchange patterns (due to the local/global model updates). Being able to detect and correctly classify FL traffic in a network is key to the realization of a robust and trustworthy FL platform as envisioned by the TRUSTroke project:

- *Quality of service:* by accurately classifying FL traffic, it is possible to prioritize the transmission of the related data, ensuring that FL training processes receive the necessary bandwidth and network resources. This optimization directly improves the FL training process, leading to faster convergence and more accurate and reliable models.
- *Client security:* FL traffic classification may be used at the clients premises as an additional measure of security in all the cases where there is low trust in the FL framework and PS. This can be achieved by deploying intelligent firewalls to enforce access control policies tailored specifically to FL-related data transmissions. This is particularly important in the context of medical clients implemented in hospitals, where security is of paramount importance.
- *Traffic shaping:* It's essential to consider the potential risks associated with malicious attacks targeting FL participants. These attackers may seek unauthorized access to stored data or attempt to disrupt FL training sessions. Traffic detection can inadvertently reveal which clients are engaged in FL activities, making them potential targets for such attacks. In response to this threat, implementing traffic shaping techniques that hinder FL traffic detection becomes crucial and enhance the overall security posture of the FL platform.

For these reasons, this paper explores the possibility of performing FL network traffic detection. Although the long-term characteristics of FL traffic are evident, we focus here on early detection, which allows for rapid reaction. Alongside classical approaches for traffic detection, the use of ML approaches resides in the complexity of the host architecture, the sensitivity of the data it stores, and the use of VPNs and Jump Host tunnelling, making the traffic to prioritize harder to identify. We have collected a detailed dataset of

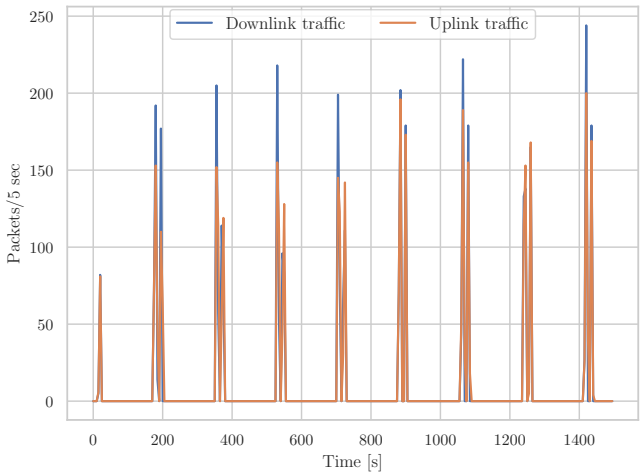

Fig. 1: Traffic from a client performing eight Federated Learning (FL) round with the Flower [5] framework

FL traffic based on Flower [5], a popular open-source FL framework, by recreating a real deployment setup with a server hosted in cloud computing platforms, and we have tested two Machine Learning (ML) approaches based on Random Forest and Neural Networks classifiers with promising results.

II. Methods

A. Dataset Creation

A dataset containing traces of FL-network traffic, as well as non-FL traces was created. For what concerns FL traffic, a testbed to generate diverse and controlled network traces was developed. The testbed allows to emulate a FL infrastructure composed of one PS and several client nodes running Flower [5], a popular and widely used FL framework known for its simplicity and prevalence in the literature [6], [7]. For what concerns the PS, we deployed it either on a Google Cloud server (2 vCPU with clock frequency of 2.25 GHz and 8 GB of RAM)or an AWS server (1 vCPU with clock frequency of 3.3 GHz and 1 GB of RAM).As on can see the two servers are characterised by different computational resources, although both were located in central Europe. As for the FL clients, they were executed via multiple containerised instances of Flower run on the same physical workstation located in Milan. This setup allowed us to simulate different FL scenarios and to capture the corresponding network traffic generated.

Multiple FL training tasks have been executed varying the number of clients, the maximum number of rounds for training the global model, the size of Neural Network (NN) model utilised, and the communication protocol used in the framework. Indeed, Flower supports three different communication protocols: gRPC request-response (gRPC rere), gRPC bidirectional-streaming (gRPC bidi), and Representational State Transfer (REST). The first two are based on HTTP/2 and Protobuf serialisation, while the latter is based on HTTP/1 and JSON payloads. Each performed experiment was either used as training or testing data, so that no overlap exist between the characteristics of the corresponding FL training tasks. Table I and II reports the details of the configurations used for the training and testing datasets. Packet-level network traffic was captured using *tcpdump* within each FL client container. To minimise bias introduced by the Linux Kernel, TCP offload has been turned off using *ethtool*. The following information are retained for each packet: packet length, timestamp, transport protocol, source and destination address, and source and destination port. In total more than 16K different TCP flows containing FL traffic have been captured.

TABLE I: Training FL dataset

Scenario	Protocol	# Clients	Model Size
1A	gRPC bidi	5/10	54.60 KB
1B		5/10	733.29 KB
2A	gRPC rere	5/10	54.60 KB
2B		5/10	733.29 KB
3A	REST	5/10	54.60 KB
3B		5/10	733.29 KB

TABLE II: Test FL dataset

Scenario	Protocol	# Clients	Model Size
1C	gRPC bidi	5/7	12.97 MB
2C	gRPC rere	6/11	12.97 MB
3C	REST	8/9	12.97 MB

For what concerns non-FL network traffic, we used a testbed architecture as close as possible to the one used for FL traffic in order to avoid bias due to the different underlying network configurations. In particular, a containerised Wi-Fi Access Point was setup with the same configuration of the FL client testbed and used to capture traffic from five laptops used by researchers during standard daily working activities for three days. Typical traffic contained in such a dataset refers to web browsing, email exchange, voice and video calls, video streaming and bulk data download. About 100K flows (both UDP and TCP) were captured for the non-FL traffic type.

B. Feature Selection and Model Description

After the dataset acquisition phase was completed, a data cleaning process was performed. This involved removing TCP acknowledgement packets, DNS packets, and flows containing fewer than two packets. Both TCP and UDP flows have been considered. Subsequently, each flow was processed with a moving window of size p packets. In this work, we used $p = 20$, which is shown as a good tradeoff between model performance and reaction time of classification in [8]. From both the captured FL and non-FL traffic we selected via undersampling about 165k windows for training and 33k windows for testing. For what concerns FL traffic, the windows are balanced among the different tested scenarios. For what concerns non-FL traffic, the training and testing windows are selected randomly. It is important to highlight that training windows were selected uniquely from traffic flows obtained when the PS was executed in the Google Cloud premises, while test windows contain both Google as well as AWS PS flows.

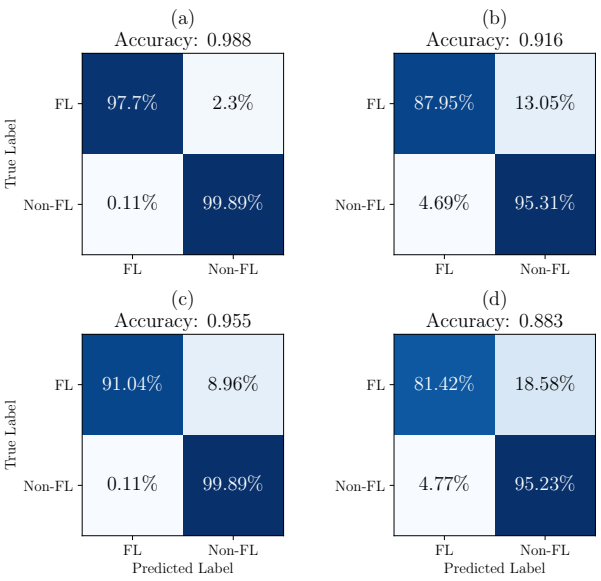

Fig. 2: Traffic Classification Performances with: RF tested with Google PS, b: NN tested with Google PS, c: RF tested with AWS PS, d: NN tested with AWS PS

We evaluate the performance of two ML models: (i) a Random Forest (RF) Classifier and (ii) a NN leveraging on Convolutional Neural Networks (CNNs) and Long Short-Term Memorys (LSTMs) networks. For what concerns the RF Classifier, we used 100 trees, no max depth limitations, minimum amount of samples to split equal to 2, minimum number of samples to be at a leaf node equal to 1, and number of features to consider when looking for the best split as $\sqrt{f}$ where f is the number of features used. The following features were extracted from each p-packet window and used in the model: packet-length (min., max., avg., std., kurtosis), inter-arrival time (min., max., avg., std., kurtosis) and packet count. The features were computed considering both the uplink and the downlink direction as well as globally. In total, $f = (5+5+1) \times 3 = 33$ features were extracted from each p-packet window. Additionally, data standardisation was applied to normalise the features. As for the NN model, we adopted the best performing architecture proposed in [8], which accepts an $n \times p$ matrix as input, where p corresponds to the number of packets considered and n to features extracted from each packet. The following (normalised) features are used in this work: frame length, frame time, inter-arrival time and a binary indicator for the direction (uplink or downlink). Therefore, $n = 5$. The detail of the utilised model, which we implemented with TensorFlow, is as it follows: *Conv1D(32,3,1,relu) - BatchNormalization - Conv1D(32,3,1,relu) - BatchNormalization - LSTM(100) - Dropout(0.2) - Flatten - Dense(100,relu) - Dropout(0.4) - Dense(2,softmax)* [8].

III. Traffic Classification Results

The two models were trained and tested over the created dataset to assess their performance. For the NN model, training was performed with 50 epochs and an early stopping criterion was adopted if the loss function didn't improve for 10 consecutive epochs. The confusion matrices and model accuracies for both models are shown in Figure 2. As one can see, both models demonstrate the capability to effectively discriminate FL network traffic from non-FL traffic, as evidenced by their accuracy in classification. In general, the RF classifier model presented better classification performance compared to its counterpart. In details, the RF classifier achieves an accuracy of 0.988 when tested with traffic from a unique combination of clients, FL rounds, and model sizes never used during the training process. Even when the PS characteristics changes (in terms of server location and performance), the RF classifier maintains a high accuracy of 0.955. In contrast, the NN model exhibits slightly lower classification performance compared to the RF classifier. When tested with the same unseen combinations of FL rounds, clients, and model sizes, the NN model achieves an accuracy of 0.916. This accuracy decreases to 0.883 when the server location and characteristics also change. We highlight that, once trained, both classifiers are able to label a flow as being of FL type after just $p = 20$ packets, therefore allowing for early detection and a quick reaction. Although FL traffic has unique and evident long-term periodical features, we deliberately decided not to use such characteristics that would delay the detection process.

IV. Conclusions

The early detection of FL traffic hold significant promise for enhancing FL performance through QoS provisioning and bolstering client security. This work addressed such a challenge by curating a dataset of FL network traffic, which served as the basis for training machine learning models for its identification. The proposed methodology enables efficient detection of FL traffic with as few as 20 packets captured from a network flow, facilitating quick response times. These findings carry substantial implications for Internet Service Providers (ISPs) and network security practitioners. ISPs can capitalize on the unique characteristics of FL traffic to implement policies that prioritize and optimize the transmission of FL model updates. However, it's crucial to acknowledge potential security risks. Attackers monitoring the network could exploit the identifiable FL traffic pattern to target clients engaged in FL activities. This underscores the need for traffic shaping techniques aimed at masking FL traffic, a topic that we plan to further investigation. Future research directions include expanding the FL dataset to evaluate the classification model's performance across different FL frameworks and configurations. As an example, we plan to study the possibility of identifying TLS-encrypted or SSH-tunneled FL traffic, which the TRUStroke project envisions as a way to enhance the robustness and security level of the proposed FL platform [9].

Acknowledgments

Funded by the EU in the call HORIZON-HLTH-2022-STAYHLTH-01-twoStage under grant agreement No 101080564.

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*, ser. Proceedings of Machine Learning Research, A. Singh and J. Zhu, Eds., vol. 54. PMLR, 20–22 Apr 2017, pp. 1273–1282. [Online]. Available: https://proceedings.mlr.press/v54/mcmahan17a.html

[2] B. S. Guendouzi, S. Ouchani, H. EL Assaad, and M. EL Zaher, "A systematic review of federated learning: Challenges, aggregation methods, and development tools," *Journal of Network and Computer Applications*, vol. 220, p. 103714, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804523001339

[3] Feb 2024. [Online]. Available: https://trustroke.eu/

[4] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 1175–1191. [Online]. Available: https://doi.org/10.1145/3133956.3133982

[5] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, J. Fernandez-Marques, Y. Gao, L. Sani, K. H. Li, T. Parcollet, P. P. B. de Gusmão *et al.*, "Flower: A friendly federated learning research framework," *arXiv preprint arXiv:2007.14390*, 2020.

[6] A. G. Samuel, S. V. Puthusseri, E. S. Eazhakadan, and M. Shetty, "Detecting malicious blockchain attacks through flower using horizontal federated learning: An investigation of federated approaches," in *2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, 2023, pp. 1–7.

[7] M. Kapsecker, D. N. Nugraha, C. Weinhuber, N. Lane, and S. M. Jonas, "Federated learning with swift: An extension of flower and performance evaluation," *SoftwareX*, vol. 24, p. 101533, 2023.

[8] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret, "Network traffic classifier with convolutional and recurrent neural networks for internet of things," *IEEE Access*, vol. 5, pp. 18 042–18 050, 2017.

[9] A. Boiano, M. D. Gennaro, L. Barbieri, M. Carminati, M. Nicoli, A. Redondi, S. Savazzi, A. S. Aillet, D. R. Santos, and L. Serio, "A secure and trustworthy network architecture for federated learning healthcare applications," 2024.