Improving detection of scanning attacks on heterogeneous networks with Federated Learning

Gustavo de Carvalho Bertoli
Aeronautics Institute of Technology
São José dos Campos, Brazil
gustavo.bertoli@ga.ita.br

Lourenço Alves Pereira Júnior
Aeronautics Institute of Technology
São José dos Campos, Brazil
ljr@ita.br

Osamu Saotome
Aeronautics Institute of Technology
São José dos Campos, Brazil
osaotome@ita.br

ABSTRACT

Scanning attacks are the first step in the attempt to compromise the security of systems. Machine learning (ML) has been used for network intrusion detection systems (NIDS) to protect systems by learning misbehavior based on network traffic. This paper demonstrates that Federated Learning (FL) is a promising approach to achieve better detection performance than traditional local training and inference on distributed agents. Also, this FL approach brings privacy, efficiency, and it is suitable for distributed ML-based NIDS solutions. We present a horizontal FL setup using Logistic Regression with `FedAvg` strategy applied to 13 agents (data silos) capable of providing an iterative process of constant learning improvement. Our results indicate a more stable learning process when observed the F1-score average, whereas the traditional NIDS approach (local trained models) present lesser performance and bigger variability to classify scanning and benign traffic. We tested our model performance on the TON_IoT dataset containing network traffic from a virtualized heterogeneous network composed of cloud, fog, and edge layers.

CCS CONCEPTS

• **Networks** → **Network security**; • **Security and privacy** → **Intrusion detection systems**; • **Computing methodologies** → **Learning settings**.

KEYWORDS

network intrusion detection, federated learning, machine learning, scanning attacks

ACM Reference Format:
Gustavo de Carvalho Bertoli, Lourenço Alves Pereira Júnior, and Osamu Saotome. 2021. Improving detection of scanning attacks on heterogeneous networks with Federated Learning. In *Proceedings of Workshop on AI in Networks and Distributed Systems (WAIN) 2021 (WAIN 2021).* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Our current network environment faces the increase of connected devices by the widespread Internet of Things (IoT) paradigm. The Internet provides connectivity to everything, from consumer devices to industrial control systems. Consequently, this increase of connected devices causes increasing network traffic and the exposure of devices to malicious agents. In this context, the need for cybersecurity is mandatory to keep these solutions operational without impacting the services and users of these systems [1, 2].

A well-known approach to comply with cybersecurity needs is the network intrusion detection systems (NIDS), which are a defense mechanism responsible for reporting security events once the matching of a known malicious behavior or anomaly from normal behavior appears on the network traffic [3]. In the NIDS domain, the use of machine learning (ML) is a well-established research path that is capable of handling the amount of network traffic generated and learn from this data to improve the security of networks [4]. However, the majority of NIDS research does not address the current network context of the huge amount of distributed systems due to the IoT context, neither the context of zero-trust architecture [5], that requires defense mechanisms on each device. In conjunction with that, we see a trend of NIDS research considering federated learning (FL) to achieve better detection performance by learning from distributed systems [6].

Furthermore, among the myriad of network attacks that is normally the focus of NIDS solutions, the specific task of detecting scanning attacks (also known as reconnaissance) is of paramount importance because the scanning is the first step of an attack effort [7, 8], and deterring this kind of attack results in stopping an attack in its early stages saving resources, and reducing impacts. To this end, our paper verifies the federated learning (FL) approach to improve NIDS by learning from distributed agents. The objective is to obtain a global model that takes advantage of the traffic diversity faced by these participants to improve the overall NIDS performance compared to the traditional approach that uses local training to detect scanning attacks. Moreover, our solution provides the same level of traditional NIDS solutions plus crucial features for the IoT solutions: privacy and continuous integration. We observed that FL improved data privacy and performance by not requiring the share of network traffic or moving these data to a central server for training. Indeed, our experimental results indicate a more stable learning process when observed the F1-score average, whereas the traditional NIDS approach (local trained models) present lesser performance and bigger variability to classify scanning and benign traffic. Also, we envision this FL architecture as a candidate to shift the NIDS deployment towards a distributed and continuous workflow as advancements of our previous proposed framework [9].

The sequence of this paper is structured as follows: Section 2 presents a review of related works about NIDS and FL, section 3 describes the methodology used on this paper to obtain the results

WAIN 2021, November 08–12, 2021, Milan, Italy

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

presented and discussed in section 4. Finally, we conclude and point out future works on section 5.

2 RELATED WORKS

This related work section discusses the current Federated Learning (FL) application research in the network intrusion detection context, focusing on its applicability to the Internet of Things (IoT) context.

The authors in [10] proposed a Federated Learning architecture using Long short-term memory (LSTM) and Gated recurrent units (GRU) for a Modbus network dataset, reporting an accuracy improvement of 4.8%. Their comparison is between the proposed FL setting with a non-FL approach that uses the classical centralized data for training. Compared to the authors, our experimental setup does not consider centralized data training once this scenario is not feasible in a distributed architecture. Hence, we evaluate local training (on each agent) and the global model after sharing the locally trained model with an external aggregator agent. The authors proposed detection of denial of service (DoS) on the IIoT context achieved the biggest F1-score improvement for 40 devices of 1.7% (GRU-2 model).

An FL setup based on attention gated recurrent unit (FedAGRU) is proposed by [11], and a reported accuracy improvement of 8%. The authors evaluated their method based on three datasets KDD-CUP99 [12], CICIDS2017 [13], and WSN-DS [14]. Their experiment is similar to the experimental design used in our methodology. They split the dataset into 10 agents and compared the performance between local training and FL setting. When comparing the best local model (SVM-GRU) and the proposed FedAGRU, there is no improvement on the F1-score. On the other hand, both IID and non-IID settings compared with local GRU-Softmax, local improved convolutional neural network (ICNN), and local variational autoencoder (VAE), resulting in an average improvement of 3%, which serves as evinces of improvements of F1-score for FedAGRU. However, it is worth mentioning that the KDD-CUP99 dataset is not representative of the current status of network traffic due to its obsolescence [15].

[16] proposed an FL architecture for Binarized Neural Networks (BNN) and deployed it on a software-defined network (SDN) scenario. This proposition uses SDN separation of the data plane, control plane, and cloud to set up this FL. The datasets used for evaluation are the CICIDS2017 [13] and ISCX Botnet 2014 [17] to support flow-level and packet-level detection, respectively. For the FL evaluation, they split the dataset from 2 to 8 domains (agents) and compared the performance between local training and the FL setting, similar to our approach. Nevertheless, the authors missed an F1-score performance comparison between local and FL approaches.

The authors of [18] propose an FL-based NIDS named Multi-view federated learning intrusion detection (MV-FLID) that uses an optimization technique for feature selection. It uses an MQTT dataset [19] to represent an IoT context under scanning and brute force attacks. This dataset presents uniflow, biflow, and packet granularity features used to train three different models that output their prediction to an ensemble model (random forest) to predict the class. Their evaluation setup considers 10 virtualized agents and ten rounds of communication between agents and aggregator. They obtained a 42% improvement on the F1-score when comparing the local versus MV-FLID approach. Compared with previous related works, this F1-score is outstanding. However, none of them uses the feature selection neither ensemble methods.

In summary, our evaluation methodology is similar to this related works, considering the local performance versus the global model using federated learning. In addition, we evidenced a common practice of providing the improvement claim based on accuracy metric instead of F1-score or other metrics applicable to the NIDS domain that addresses the typical imbalance characteristic of this domain. Another characteristic is the variety of datasets used to validate the approaches that make difficult the comparison between different techniques and datasets.

3 METHODOLOGY

In this methodology section, we present the rationale behind the chosen dataset. We explain the required preprocessing for our experiment and how we designed the experiments to evaluate federated learning for the specific classification task between benign and scanning traffic.

3.1 TON_IoT dataset

In this paper, we chose the TON_IoT dataset [20] to validate our research question. It represents a recent dataset with the current heterogeneity characteristics of the networks. This heterogeneity embraces technological aspects such as cloud computing, and those related to the Internet of Things (IoT) paradigm, presenting devices in an architectural scheme in the fog and edge layers. The dataset is composed of benign traffic of IoT and IIoT devices. On the other hand, the attack traffic is composed of 9 attacks performed by a virtual machine in this testbed considering the following attacks: scanning, distributed and traditional denial of service (DoS), ransomware, backdoor, injection attack, cross-site scripting (XSS), password cracking, and man-in-the-middle (MITM).

Our methodology considers the learning task as a binary classification between the normal and scanning traffic, following the recommendation of *keeping the scope narrow* for ML in NIDS [15]. Nmap and Nessus vulnerability scanners were responsible for generating the scanning attacks. The 5 source IP addresses of these attacks are 192.168.1.30, 192.168.1.31, 192.168.1.32, 192.168.1.33, 192.168.1.38; this definition supports easy labeling and understanding of the dataset. Despite this dataset generation using two specific tools for scanning attacks, we understand that this learning task applies to similar attacks as reported by [21] that demonstrates the capability of learning from known attacks and satisfactory performance to similar attacks.

Regarding TON_IoT, we understand the issues of features commonality between NIDS datasets, making the deployment of ML-based NIDS difficult or performance comparison between diverse datasets difficult. To address this concern, we use a modified version of the TON_IoT dataset that processes the original dataset with NetFlow to generate 43 standard network flow features, named by the authors as NF-ToN-IoT-v2 [22].

3.2 Dataset Preprocessing

The NF-ToN-IoT-v2 is a 5.7GB dataset composed of 45 columns which 43 represents the features obtained with NetFlowV2, and

two extra columns named attack and label. The attack column describes if the specific flow is one of the ten classes (benign or one out of nine specific attacks). Label corresponds to a binary classification of the benign and attack flows (0 and 1, respectively).

The first step in our preprocessing stage was to filter only network traffic with the benign and scanning attack label. This decision is to comply with our experiment design. This preprocessing step reduced the original dataset to 3.3GB, with 9.880.938 samples with 62% benign and 38% scanning traffic. Then, we evaluated how many targets (i.e., IP destination address) face the Scanning attacks. This evaluation uses the five source IP addresses used by VM to perform scanning attacks. This evaluation results in 346 different targets. However, we decided to move forward with targets that contains more than 3.000 network traces, resulting in 229 targets.

The next preprocessing step follows these 229 targets. First, we evaluate the imbalance characteristics between benign and attack samples. The majority of these targets present a class imbalance of around 98% of attack traffic (approx. 3000 samples) and about 2% of benign traffic (approx. 70 samples). Based on domain knowledge, most traffic in the NIDS context is imbalanced, but with the majority of benign traffic. Therefore, we decided to consider those targets with a fair quantity of benign traffic. This decision reduced the scope from 229 to 13 targets.

The majority of features obtained with NetFlowV2 are integer and floating points. It requires that a correct value replace the values not complying with these formats. Thus, it is part of the preprocessing step to replace not a number (*NaN*) by 0, and infinite values (*inf*) by 9999.

We removed from the dataset the attack column because we can rely on the binary "label" column, that after previous steps will represent just benign and scanning samples. Also, it removes the protocol's features, and for both source and destination, the IP addresses and port numbers. This decision is to avoid the learning task resulting in a model that describes the testbed architecture or the dataset itself instead of benign and scanning traffic behavior. As an example, [23] reports high feature importance for attributes such as source/destination IP addresses, source/destination ports, and protocol, which do not allow to use of this prior knowledge (source/destination addresses) to deploy the solution on the real operational setting.

3.3 Logistic Regression

The machine learning algorithm used in our paper is the logistic regression, using the stochastic gradient descent (SGD) classifier available on the scikit-learn library citescikit with Log loss. This decision was based on the algorithm's simplicity and good performance obtained during preliminary tests on the local data. From a dataset $\mathcal{D}$ composed by m samples, with x representing a set of n features, and y the class label for each example:

$$\mathcal{D} = \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), ..., (x^{(m)}, y^{(m)})\} \tag{1}$$

We train a Logistic Regression algorithms capable to predict the class label $\hat{y} = P(y = 1|x)$ given an input vector x, with $x \in \Re^n$. The parameters of the Logistic Regression are the weights $w \in \Re^n$ associated with each input feature, and its bias (or intercept) $b \in \Re$. The $\hat{y}$ is the value obtained from sigmoid function (σ) applied to the traditional linear regression ($w^T x + b$). The log loss is used as loss function ($\mathcal{L}$) for the stochastic gradient descent (using learning rate and derivative of cost function), and to obtain the w and b parameters, a cost function ($\mathcal{J}$) is calculated over the training samples ($x^{(i)} \in \Re^n, y^{(i)}$) from training dataset of m samples:

$$\hat{y} = \sigma(w^T x + b) \tag{2}$$

$$\mathcal{L}(\hat{y}, y) = -(y \log \hat{y} + (1 - y) \log(1 - \hat{y})) \tag{3}$$

$$\mathcal{J}(w, b) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(\hat{y}^{(i)}, y^{(i)}) \tag{4}$$

The simplicity of this parametric algorithm based on weights (w) and bias (b) allows a simpler implementation of federated learning using the *federated average* (`FedAvg`).

3.4 Federated Learning (FL) Architecture

Starting from the preprocessed dataset, we design the experiment to evaluate the improvement of a global model taking advantage of federated learning (FL), in contrast, with the traditional local model training. We design this experiment as a horizontal federated learning setup, with each of the 13 targets composing 13 different data silos. These silos represent all traffic that was targeted to the specified destination IP address of these targets. Now on, we refer to the targets as agents, as each of these 13 data silos is agents performing local data training and sharing their local model for global aggregation.

The premise of our experiment in this horizontal FL setup is that all agents use the same feature set derived from NetFlowV2 without the source/destination IP, ports, and protocol. Also, each agent has access to only its traffic, complying with horizontal FL set up; i.e., with the same feature space but different sample space between agents [24]. Figure 1 illustrates of the horizontal FL setup.

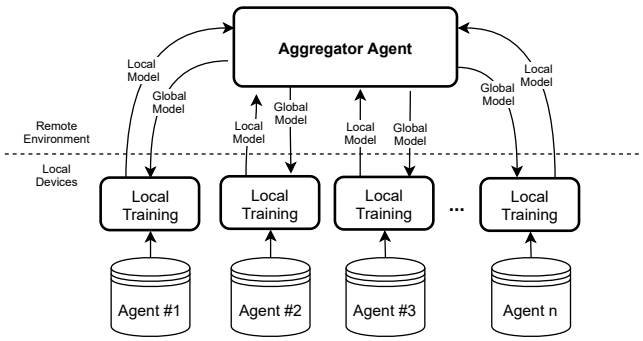

Figure 1: Horizontal Federated Learning (FL). Multiples agents perform local training on their data and share the trained model with a central entity that aggregates these models and returns a global model to the agents participating in the federated learning scheme.

Regarding the learning metric, it is important not to rely on accuracy as the evaluation metric due to NIDS's inherent imbalance characteristic. Our analysis is based on the F1-score, which is a weighted average of both precision and recall. Precision counts for correct predictions (true positives - tp) discounted by false positives

(fp), Recall counts the correct predictions (tp) while discounting false negatives (fn):

$$\text{precision} = \frac{tp}{tp + fp} \tag{5}$$

$$\text{recall} = \frac{tp}{tp + fn} \tag{6}$$

$$\text{F1-score} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{7}$$

For this FL setup, Algorithm 1 lists the overall algorithm with a global logistic regression model with zeroed initialized coefficients (one for each feature) and the intercept parameter. The stopping criteria for our FL setup use the number of rounds. On each round, all agents train locally, and the global model is updated using FedAvg. This local train of agents consists of copying the current global model to each agent participating in the FL (in our case, 13); in sequence, selecting a random subset of the agent's siloed data according to the pre-defined batch size. Then, the local agents are optimized based on this subset data and the pre-defined epoch. After this local optimization of all agents, they are aggregated into the global model using the `FedAvg` strategy with a weighted average of agents' parameters (coefficients and intercepts), and this weights using a pre-defined strategy.

ALGORITHM 1: Horizontal Federated Learning using the `FedAvg` aggregation method using number of rounds as stop criteria.

```
Input: rounds, batch size, epochs, learning rate, agents
Output: global model
global_coefficients = 0
global_intercept = 0
for n <= number of rounds do
    for agent in agent[1..13] do
        agent ← global_model
        subset = retrieve_data(agent_silo, batch_size)
        agent ← optimize(agent, subset, epochs)
    end
    weights ← calculate_weights
    global_model ← FedAvg(agent[1..13], weights)
end
```

4 RESULTS AND DISCUSSION

Based on our experiment described in section 3, we confirmed a better performance of the Logistic Regression in a Federated Learning (FL) setup with `FedAvg` approach, when compared with the traditional local training, and test on these data silos.

We simulated an FL scenario with 50 rounds and set the Logistic Regression (SGD - Log loss) parameters for 10 epochs, for a batch size of 100 samples from each silo, and a learning rate of 0.15, with a total of 13 agents (sample size). This configuration results in a mean F1-score for local agents of 0.84 ± 0.03, and a mean F1-score when evaluating the `FedAvg` global model on the 13 agents of 0.85 ± 0.02; it means a better F1-score capable of providing the same level of threat identification, but with additional crucial features for the IoT domain (e.g., privacy and distributed learning). A statistical T-test confirms this improvement with statistical significance for p<0.01 (p=0.003). The results is illustrated by the F1-score distribution on Figure 2.

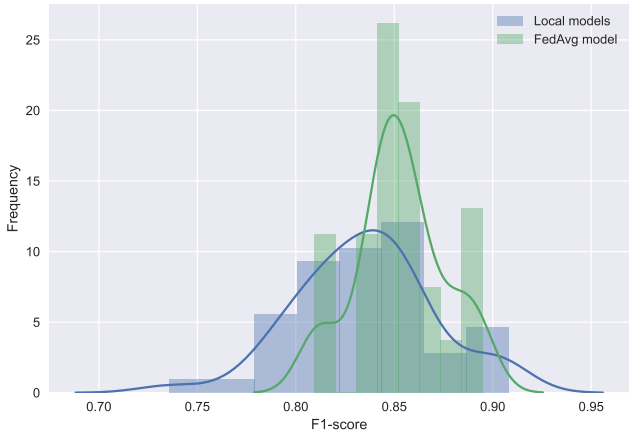

Figure 2: Average F1-score distribution using Logistic Regression. `FedAvg` global model and Local models applied to the 13 agents/silos (rounds=50, epochs=10, batch size=100, learning rate=0.15, agents=13).

We introduced an additional preprocessing step that downsamples the majority class of each silo before horizontal FL processing to achieve these results. Additionally, we obtained an improvement of `FedAvg` algorithm, using a weighted average based on the original imbalance characteristics of each silo before the downsampling. This weight calculation is based on equation 8 applicable to each silo, with total samples representing the agent's dataset size, the number of classes (*n_classes*) equal 2 representing just benign and scanning, and the *scanning_samples* representing the total samples of scanning traffic on the respective agent's dataset:

$$w_{scanning} = \frac{total\_samples}{n\_classes \cdot scanning\_samples} \tag{8}$$

The summary of each agent, the IP address from defined on the TON_IoT dataset, the description of this device and the respective layer on the heterogeneous architecture, in conjunction with the original silos dataset size (silo samples), their imbalance characteristics between benign and scanning traffics, the calculated weight for each agent contribution during `FedAvg`, and both local and `FedAvg` F1-score applicable to each agent/silo is summarized on the Table 1.

Additionally to the average increase of the F1-score performance (mean of the 13 agents), we present an agent-by-agent analysis based on F1-score for this experiment:

- Agents with FL better than Local: 1, 2, 3, 7, 10, 11, 12
- Agents with Local better than FL: 4, 6, 13
- Agents with same performance for FL and Local: 5, 8, 9

We can highlight an expressive F1-score improvement using FL for agents 7 (+13%) and 11 (+16%). On the other hand, Agent 4 (−12%) and 6 (−16%) present a poor performance when comparing the global model with their local training.

Table 1: Summary table containing the descriptive information of each agent, its siloed data, and the obtained F1-score in the Federated Learning setup using `FedAvg`, and local training.

	IP address	Description	Silo Samples	% Benign	% Scanning	Scanning Class Weight	FedAvg F1-Score	Local F1-Score
Agent 1	192.168.1.152	Device (Edge)	565272	49.9	50.1	1.0	0.94	0.88
Agent 2	192.168.1.193	Windows 7 (Fog)	499300	67.5	32.5	1.54	0.93	0.91
Agent 3	192.168.1.190	Orchestrated Server (Fog)	962313	64.8	35.2	1.42	0.86	0.79
Agent 4	192.168.1.1	Router (Edge)	68058	94	6	8.35	0.64	0.73
Agent 5	192.168.1.180	Security Onion (Fog)	1288529	56	44	1.14	0.98	0.98
Agent 6	192.168.1.49	Device (Edge)	1016646	98.4	1.6	32.03	0.32	0.38
Agent 7	192.168.1.194	Metasploitable 3 (Fog)	528613	56.2	43.8	1.14	0.88	0.78
Agent 8	192.168.1.46	Device (Edge)	454179	77.4	22.6	2.21	0.97	0.97
Agent 9	192.168.1.186	Device (Edge)	457054	72.3	22.7	1.81	0.97	0.97
Agent 10	192.168.1.195	Windows 10 (Fog)	1201419	51.7	48.3	1.03	0.95	0.94
Agent 11	192.168.1.169	Device (Edge)	730266	45.4	54.5	0.92	0.89	0.77
Agent 12	192.168.1.133	Device (Edge)	68344	61.3	38.7	1.29	0.92	0.89
Agent 13	192.168.1.79	Device (Edge)	310574	83.2	16.8	2.98	0.83	0.85

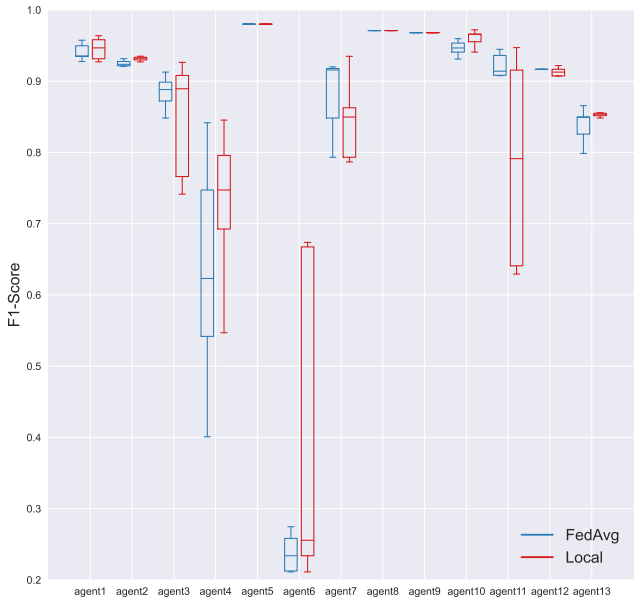

Figure 3: F1-score comparison between Federated Learning (FedAvg) and Local training for each of the 13 agents under analysis (rounds=50, epochs=10, batch size=100, learning rate=0.15, agents=13).

Figure 3 provides a visualization of the F1-score for each agent during the simulated 50 rounds for both Federated Learning (`FedAvg`) and Local training. From this figure, it is possible to visualize the high range of F1-score for the Local training (also evidenced by Local F1-score distribution in Figure 2), except for Agent 4 that presents a high range for `FedAvg` tests. Furthermore, we can see that the federated approach yielded less variance in the performance, meaning better learning and attack detection.

Based on the results on the Table 1, we evidenced a negative correlation between the *Scanning Class Weight* and both `FedAvg` (−0.95), and Local (−0.85) F1-scores. This negative correlation raises a further research question to be evaluated: How *% Scanning* imbalances affect the Federated Learning setting? What strategies could address it?

It is important to point out that additional evaluations are needed, such as a grid search for a variable number of agents, variable batch sizes, other options of epochs, and the number of rounds. Based on the TON_IoT dataset, and specifically the Scanning *vs.* Benign subsets (our agents' silos), we obtained a successful FL improvement compared to local training, despite the Non-IIDness of the data. These Non-IID characteristics are evidenced by the label distribution skew between silos comparing Agent 1 with Agent 6 and the quantity skew (i.e., samples) between silos as confirmed by Agent 4's silo samples versus Agent 5's silo samples. This characteristic opens up other research directions to address those Non-IID characteristics of the data, to validate the FL performance increase [25, 26]. These Non-IID characteristics are expected on real-world network traffic, and its evaluation is important to raise the requirements for a successful deployment of distributed NIDS solutions.

Regarding the concerns to bridge the gap to real-world application, we understand that logistic regression is a parametric algorithm that is a good solution for resource-constrained devices as reported by [9]. It reports for logistic regression a reduced storage usage, fast inference time, and low CPU and RAM usage compared to other algorithms such as k-nearest neighbors, tree-based models, or multi-layer perceptron. However, despite our methodology similar to those reported in the section 2 which compares local trained models to federated learning approach for distributed ML-based NIDS, for this benign versus scanning learning problem, we report a 0.93 F1-score when considering the traditional centralized data training approach, that is unfeasible for real-world application.

Also, about practical considerations for realistic deployment, the dataset uses flow features extracted from network traffic (using NetFlow), so it is expected that the major contribution in a federated learning setting is the sharing of malicious behavior between agents participating in the federation after an attack is reported in one of

them. Therefore, in this realistic deployment, an active learning setting or generative models must be in place for labeling the events during operation.

5 CONCLUSION

This paper presented a horizontal federated learning approach to support the improvement of Network Intrusion Detection Systems (NIDS). This approach was validated by an experiment using a recent dataset (TON_IoT) that represents a heterogeneous network similar to the network's current characteristics that rely on edge, fog, and cloud layers. This experiment produced a distributed learning process capable of providing less variability for F1-score performance without further data handling to address the non-IID characteristics. Therefore, we observed that our approach propitiates adding new crucial features to NIDS solutions (such as privacy, distributed learning) and guarantees the same level of threat identification observed in the traditional NIDS approach.

As future work, we plan to explore different machine learning algorithms, techniques (feature selection and ensembles), aggregation strategies, and investigates other methods to support the continuous generation of up-to-date datasets for this distributed NIDS using federated learning gains as advancements of our proposed framework [9].

We provide online resources and the source code to reproduce this paper is available on our online repository https://github.com/c2dc/wain2021.

ACKNOWLEDGEMENTS

This work was supported in part by ITA's Programa de Pós-graduação em Aplicações Operacionais (ITA/PPGAO).

REFERENCES

[1] Tianlong Yu, Vyas Sekar, Srinivasan Seshan, Yuvraj Agarwal, and Chenren Xu. Handling a trillion (unfixable) flaws on a billion devices: Rethinking network security for the internet-of-things. In *Proceedings of the 14th ACM Workshop on Hot Topics in Networks*, pages 1–7, 2015.

[2] Francesca Meneghello, Matteo Calore, Daniel Zucchetto, Michele Polese, and Andrea Zanella. Iot: Internet of threats? a survey of practical security vulnerabilities in real iot devices. *IEEE Internet of Things Journal*, 6(5):8182–8201, 2019.

[3] Nadia Chaabouni, Mohamed Mosbah, Akka Zemmari, Cyrille Sauvignac, and Parvez Faruki. Network intrusion detection for iot security based on learning techniques. *IEEE Communications Surveys Tutorials*, 21(3):2671–2701, 2019.

[4] Ansam Khraisat, Iqbal Gondal, Peter Vamplew, and Joarder Kamruzzaman. Survey of intrusion detection systems: techniques, datasets and challenges. *Cybersecurity*, 2(1):20, 2019.

[5] Scott Rose, Oliver Borchert, Stu Mitchell, and Sean Connelly. Zero trust architecture. Technical report, National Institute of Standards and Technology, 2019.

[6] Sawsan Abdul Rahman, Hanine Tout, Chamseddine Talhi, and Azzam Mourad. Internet of things intrusion detection: Centralized, on-device, or federated learning? *IEEE Network*, 34(6):310–317, 2020.

[7] Tarun Yadav and Arvind Mallari Rao. Technical aspects of cyber kill chain. In *International Symposium on Security in Computing and Communication*, pages 438–452. Springer, 2015.

[8] Blake E Strom, Andy Applebaum, Doug P Miller, Kathryn C Nickels, Adam G Pennington, and Cody B Thomas. Mitre att&ck: Design and philosophy. *Technical report*, 2018.

[9] Gustavo De Carvalho Bertoli, Lourenço Alves Pereira Júnior, Osamu Saotome, Aldri L. Dos Santos, Filipe Alves Neto Verri, Cesar Augusto Cavalheiro Marcondes, Sidnei Barbieri, Moises S. Rodrigues, and José M. Parente De Oliveira. An end-to-end framework for machine learning-based network intrusion detection system. *IEEE Access*, 9:106790–106805, 2021.

[10] Viraaji Mothukuri, Prachi Khare, Reza M. Parizi, Seyedamin Pouriyeh, Ali Dehghantanha, and Gautam Srivastava. Federated learning-based anomaly detection for iot security attacks. *IEEE Internet of Things Journal*, 2021.

[11] Zhuo Chen, Na Lv, Pengfei Liu, Yu Fang, Kun Chen, and Wu Pan. Intrusion detection for wireless edge networks based on federated learning. *IEEE Access*, 8:217463–217472, 2020.

[12] S Hettich. Kdd cup 1999 data. *The UCI KDD Archive*, 1999.

[13] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A Ghorbani. Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSp*, 1:108–116, 2018.

[14] Iman Almomani, Bassam Al-Kasasbeh, and Mousa Al-Akhras. Wsn-ds: A dataset for intrusion detection systems in wireless sensor networks. *Journal of Sensors*, 2016, 2016.

[15] Robin Sommer and Vern Paxson. Outside the closed world: On using machine learning for network intrusion detection. In *2010 IEEE Symposium on Security and Privacy*, pages 305–316, 2010.

[16] Qiaofeng Qin, Konstantinos Poularakis, Kin K Leung, and Leandros Tassiulas. Line-speed and scalable intrusion detection at the network edge via federated learning. In *2020 IFIP Networking Conference (Networking)*, pages 352–360. IEEE, 2020.

[17] Elaheh Biglar Beigi, Hossein Hadian Jazi, Natalia Stakhanova, and Ali A. Ghorbani. Towards effective feature selection in machine learning-based botnet detection approaches. In *2014 IEEE Conference on Communications and Network Security*, pages 247–255, 2014.

[18] Dinesh Chowdary Attota, Viraaji Mothukuri, Reza M. Parizi, and Seyedamin Pouriyeh. An ensemble multi-view federated learning intrusion detection for iot. *IEEE Access*, 9:117734–117745, 2021.

[19] Hanan Hindy, Ethan Bayne, Miroslav Bures, Robert Atkinson, Christos Tachtatzis, and Xavier Bellekens. Machine learning based iot intrusion detection system: an mqtt case study (mqtt-iot-ids2020 dataset). In *International Networking Conference*, pages 73–84. Springer, 2020.

[20] Abdullah Alsaedi, Nour Moustafa, Zahir Tari, Abdun Mahmood, and Adnan Anwar. Ton_iot telemetry dataset: A new generation dataset of iot and iiot for data-driven intrusion detection systems. *IEEE Access*, 8:165130–165150, 2020.

[21] Eduardo K. Viegas, Altair O. Santin, and Luiz S. Oliveira. Toward a reliable anomaly-based intrusion detection in real-world environments. *Computer Networks*, 127:200–216, 2017.

[22] Mohanad Sarhan, Siamak Layeghy, Nour Moustafa, and Marius Portmann. Towards a standard feature set of nids datasets. *arXiv preprint arXiv:2101.11315*, 2021.

[23] Nour Moustafa. A new distributed architecture for evaluating ai-based security systems at the edge: Network ton_iot datasets. *Sustainable Cities and Society*, 72:102994, 2021.

[24] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19, 2019.

[25] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.

[26] Hangyu Zhu, Jinjin Xu, Shiqing Liu, and Yaochu Jin. Federated learning on non-iid data: A survey. *arXiv preprint arXiv:2106.06843*, 2021.