Federated Learning for Zero-Day Attack Detection in 5G and Beyond V2X Networks

1st Abdelaziz Amara korba
L3I, University of La Rochelle, France
LRS, Badji Mokhtar Annaba University, Algeria

2nd Abdelwahab Boualouache
FSTM, University of Luxembourg
Luxembourg

3rd Bouziane Brik
DRIVE, University of Bourgogne Franche Comte, France

4rd Rabah Rahal
LRS, Badji Mokhtar Annaba University, Algeria

5rd Yacine Ghamri-Doudane
L3I, University of La Rochelle,France

6rd Sidi Mohammed Senouci
DRIVE, University of Bourgogne Franche Comte, France

***Abstract*—Deploying Connected and Automated Vehicles (CAVs) on top of 5G and Beyond networks (5GB) makes them vulnerable to increasing vectors of security and privacy attacks. In this context, a wide range of advanced machine/deep learning-based solutions have been designed to accurately detect security attacks. Specifically, supervised learning techniques have been widely applied to train attack detection models. However, the main limitation of such solutions is their inability to detect attacks different from those seen during the training phase, or new attacks, also called zero-day attacks. Moreover, training the detection model requires significant data collection and labeling, which increases the communication overhead, and raises privacy concerns. To address the aforementioned limits, we propose in this paper a novel detection mechanism that leverages the ability of the deep auto-encoder method to detect attacks relying only on the benign network traffic pattern. Using federated learning, the proposed intrusion detection system can be trained with large and diverse benign network traffic, while preserving the CAVs' privacy, and minimizing the communication overhead. The in-depth experiment on a recent network traffic dataset shows that the proposed system achieved a high detection rate while minimizing the false positive rate, and the detection delay.**

***Index Terms*—5GB, Connected and Automated Vehicles, Security, Zero-day attacks, Federated Learning**

I. Introduction

Fifth generation (5G) and beyond (5GB) networks promise to revolutionize the transportation industry by enabling ultra-reliability with ultra-low latency and high bandwidth communications [1]. These advances will significantly empower many verticals, such as smart agriculture, health, and Intelligent Transportation Systems (ITS). As part of ITS, Connected and Automated Vehicles (CAVs) have been taken significant and careful considerations in 3GPP 5G standards [2]. Specifically, integrating V2X communications into the 5G ecosystem has enabled innovative use cases and applications, such as advanced driving, vulnerable road user protection, and vehicle platooning [3]. Yet this progress is expected to be extended with 5GB, contributing thus to reducing traffic accidents and dramatically saving road users' lives. However, CAVs at all automation levels will face a massive vector of cyberattacks coming from 5GB technologies and leading to hazardous situations for road users. For example, Distributed and Denial of Service (DDoS) attacks have already been demonstrated to break 5G services [4]. But the impact of these attacks are likely to be more expansive with the integration of CAVs. More than this, cyberattacks are working continuously to develop novel tactics for breaching and breaking such systems. Facing all these challenges, Machine Learning (ML) appears as a key cybersecurity enabler to protect 5GB-enabled CAVs [5]. Various Machine Learning (ML)/ Deep Learning (DL) based Intrusion Detection Systems (ML/DL-based IDSs) have been proposed to protect vehicular networks against attacks. Most of them rely on supervised and centralized learning [6]. Centrally training the detection model requires significant data collection and labeling, which increases the communication overhead, and may raise privacy concerns. To mitigate centralized learning limitations, collaborative ML [7] has been used, enabling thus continuous accuracy evolution and flexibility. Nevertheless, several limitations exist in early collaborative ML-based IDSs [8]–[10]. Specifically, they generate a significant communication overhead during ML model updates and may violate data privacy, since learning nodes might share private information. To cope with the aforementioned issues, recent research [11]–[14] leveraged the potential of federated leaning (FL) paradigm, which has shown promising results in many applications. FL is a distributed ML paradigm allowing several nodes to train a global model cooperatively without sharing their datasets, avoiding thus overhead and mitigating privacy risks [7]. Interestingly, all existing FL-based IDSs for 5GB-enabled CAVs rely on supervised learning techniques. One important limitation of using such techniques is their inability to detect attacks different from those seen during the training phase (unseen attacks), and zero-day attacks. Another challenging issue is data imbalance, i.e., the numbers of benign and malicious traffic samples are not in the same range. Benign network traffic samples are easily available. On the other hand, malicious samples are scarce or unavailable. The lack of a thorough dataset of attack samples limits the usage of supervised techniques. Finally, most existing detection approaches assume that FL clients maintain labeled datasets that may use at each round. This assumption may not be realistic, as CAVs cannot label the network flow on every turn.

As a CAV runs a set of well-known applications (safety, convenience, commercial, etc), their communication pattern

should present a high degree of regularity so long as they are not under attack or faulty. Similarly, an attack must alter its communication pattern. Therefore, we believe it is possible to use anomaly detection techniques to model the CAV's benign (or expected) communication pattern and detect attacks as anomalous occurrences. To overcome the aforementioned limitations of the existing IDSs, in this paper, we propose an unsupervised federated learning based IDS that leverages a deep auto-encoder model to train the detection model, relying only on benign network traffic. Thanks to federated learning, the proposed IDS can be trained with large and diverse benign network traffic, while preserving the CAVs' privacy. The proposed IDS aggregates the detection model updates within the Multi-access Edge Computing (MEC) server to enhance the learning efficiency and minimize latency. The in-depth experiment on a recent network traffic dataset shows that the proposed system achieves a high detection rate while minimizing the false positive rate, and the detection delay.

The remainder of this paper is organized as follows. Section II describes related work. The design of our scheme is presented in Section III. Section IV depicts the performance evaluation results and finally, Section V concludes the paper.

II. Related Work

Several distributed ML-based IDSs have been proposed for detecting attacks in 5GB-enabled CAVs. The authors of [8] presented a DL-based IDS to detect anomalies in ITS based on Long Short-Term Memory (LSTM). In this work, time series data are collected by CAVs and sent to the cloud to enable the training and retraining of a global model using a cluster of servers instead of one server. The authors of [9] proposed a collaborative IDS based on supervised DL, Generative Adversarial Networks (GANs), and Software Defined Networking (SDN). The proposed system enables distributed SDN controllers managing sub-networks of CAVs to train a global model for the whole network without directly exchanging their sub-network flows. However, both [8] and [9] raise privacy issues since datasets are shared between learning nodes. The authors of [10] proposed a distributed ML-based IDS that enables CAVs to directly communicate to train a global model based on supervised learning without sharing their datasets. However, peer-to-peer distributed learning generates a large overhead, degrading communication performance. The authors of [11] proposed an FL-based privacy-preserving collaborative IDS for CAVs. This work enables CAVs (FL clients) to train DL models on a locally labeled dataset and share their parameters with the central FL server to build a global model. The authors of [12] proposed a FL-based privacy-preserving collaborative IDS based on supervised learning that leverages a set of FL servers to train the global model. The authors of [13] proposed FL for collaborative IDS. This work suggests offloading the training to distributed vehicular edge nodes. Specifically, CAVs act as FL clients for building models based on their locally labeled datasets and Roadside Units (RSUs) for aggregating global models. The authors in [14] proposed SDN-FL-based IDS for CAVs. In this work, SDN controllers train local models based on labeled datasets built using data collected from CAVs, while the aggregation of global models is performed on the cloud.

Overall, existing FL-based IDSs for 5GB-enabled CAVs [11]–[14] have specifically two main limitations: (i) they are based on supervised learning which limits their effectiveness against unseen and zero-day attacks, and (ii) they assume that FL clients have labeled datasets, which in practice might be unrealistic. Considering these gaps, we propose a novel network-based IDS trained using a deep auto-encoder model. Relying only on benign network traffic, our system can detect unseen or zero-day attacks so long as they alter the benign communication pattern of the CAV. Additionally, our solution does not compromise the CAV's privacy since it is built through federated learning.

III. Proposed Solution

Collaborative learning allows training the model with a large amount of network traffic from diverse CAVs, while preserving data privacy, and minimizing the communication overhead. The proposed MEC-enabled learning scheme trains the deep Auto-Encoder (AE) model in a federated way, as illustrated in Figure 1. First, the raw captured packets are converted to flows. Then, for each flow, a set of pertinent features are calculated. Next, the local dataset of flows is fed to the AE model initially distributed by the MEC server. The training rounds are orchestrated by the MEC server and executed by the AE on the participating CAV's local dataset.

A. Flow extraction & features engineering

First, to identify a traffic flow, we use a combination of five properties from the packet header, including the network and the transport layer headers of the TCP/IP protocol stack. These are as follows: source IP address, destination IP address, source port number, destination port number, and protocol. For each flow extracted, a set of features are calculated according to a given time window (ex. 100 seconds). Flow features include mainly packet header characteristics and statistics computed from the aggregating network and transport layers header information of the packets in a flow. This set of features is then used in the form of a features vector. The network features can be categorized into Time, Inter-arrival, PacketsBytes, and Flags groups. The list of features as well as their description are given in table I.

B. Modeling benign network traffic pattern

The Auto-Encoder (AE) model [15] is an unsupervised model that compresses input vectors as code vectors using a set of recognition weights and then converts back to m $(m < d)$ number of neurons reconstructed input vectors using a set of generative weights. There are two major parts in an AE architecture: the encoder and the decoder. The encoder reduces the dimension of the input vectors ($x_i \in R^d$) to numbers of neurons that form the hidden layer. The activation of the neuron i in the hidden layer is given by:

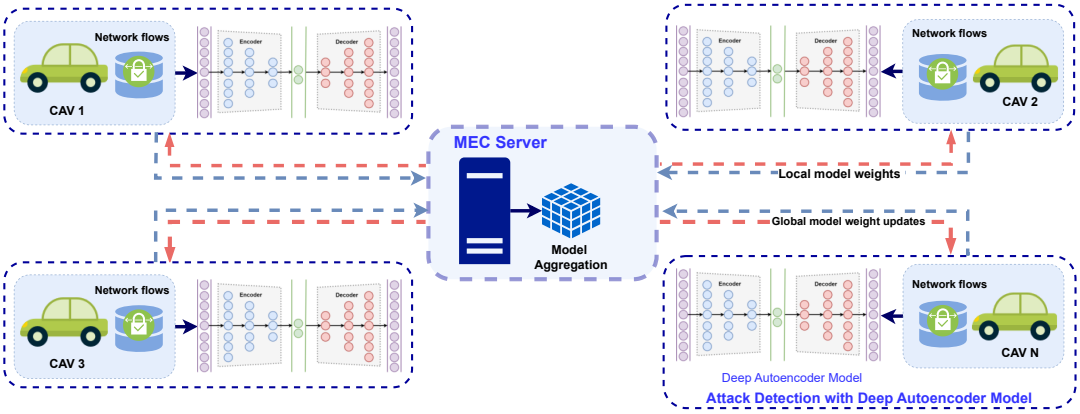

Fig. 1: MEC-Enabled Federated learning architecture

TABLE I: List of network features

Feature names	Description
numHdrs	Number of headers (depth) in hdrDesc
l4Proto	Layer 4 protocol (TCP, UDP, HOPOPT,...)
dstPortClass	Port based classification of the destination port name (HTTPS, Telnet, NTP, ..)
numPktsSnt, numByteSnt	Number of transmitted packets/ byte per second
minPktSz, maxPktSz, avePktSize, stdPktSize	Minimum/Maximum/Average/ Standard deviation of layer 3 packet size
pktAsm, bytAsm	Packet/ Byte stream asymmetry
duration	Duration of the flow in seconds
minIAT, maxIAT, aveIAT, stdIAT	Minimum/ Maximum/ Average/ Standard deviation of inter-arrival time
pktps/ bytps	Number of sent packets/ byte per second
ipTTLChg, ipTOS, ipFlags	IP TTL change count, IP Type of Service, IP aggregated flags
tcpISeqN	TCP Initial Sequence Number
tcpSeqFaultCnt	TCP sequence number fault count
tcpPAckCnt	TCP packet ack count
tcpFlwLssAckRcvdBytes	TCP flawless ack received bytes
tcpInitWinSz, tcpAveWinSz, tcpMinWinSz, tcpMaxWinSz,	TCP initial/ average / minimum/ Maximum effective window size
tcpWinSzDwnCnt, tcpWinSzUpCnt	TCP effective window size change down/up count
tcpWinSzThRt	TCP packet count ratio below window size WINMIN threshold
tcpFlags	TCP aggregated protocol flags (cwr, ecn, urgent, ack, push, reset, syn, fin)
tcpAnomaly	TCP aggregated header anomaly flags
tcpOptions	TCP aggregated options
tcpMSS	TCP maximum segment size
tcpEcI	TCP estimated counter increment
tcpUtm, tcpBtm	TCP estimated up/boot time
tcpSSASAATrip	TCP trip time SYN, SYN-ACK Destination — SYN-ACK, ACK Source
tcpRTTAckTripMin, tcpRTTAckTripMax, tcpRTTAckTripAve	TCP ACK minimum/ maximum/ average trip time
tcpRTTAckTripJitAve, tcpRTTAckJitAve	TCP ACK trip/ round trip time average jitter
tcpRTTSseqAA	TCP round trip time {SYN, SYN-ACK, ACK} and {ACK-ACK}

$$h_i = f^{\theta}(x) = s(\sum_{j=1}^{n} W_{ij}^{input} x_j + b_i^{input}) \quad (1)$$

where x is the input vector, θ is the parameters $\{W^{input}, b^{input}\}$, W is an encoder weight matrix of dimension $m \times d$, while b is a bias vector of dimension m. Thus, the input vector is encoded to a vector with fewer dimensions. The decoder maps the low-dimensional hidden representation h_i to the original input space R^d by the same transformation as the encoder. The function of mapping is as follows:

$$x_{'}^{i} = g_{\theta'}(h) = s(\sum_{j=1}^{n} W_{ij}^{hidden} h_j + b_i^{hidden}) \quad (2)$$

The set of decoder parameters is $\theta'(W^{hidden} h_j + b^{hidden})$. The objective of an autoencoder is to minimize the reconstruction error relative to θ and θ' :

$$\theta^*, \theta^{'*} = arg_{\theta,\theta'} min \frac{1}{n} \sum_{i=1}^{n} \varepsilon(x_i, x_i^{'}) \quad (3)$$

$$= arg_{\theta,\theta'} min \frac{1}{n} \sum_{i=1}^{n} \varepsilon(x_i, g_{\theta'}(f_{\theta}(x_i))) \quad (4)$$

The reconstruction error is utilized as the anomaly score. Network flows with significant reconstruction errors are regarded as malicious flows (anomalies). Only benign flows are used to train the AE model. After training, the AE model will reconstruct benign flows exceptionally well, but not malicious flows that it has never seen. **Algorithm** 1 shows the anomaly detection process using the reconstruction errors of the AE model. As shown in equation 5, the threshold, $alpha$, is the sum of the sample's mean squared error (MSE) median and the sample's five times the MSE median absolute deviation

(MAD) over the validation set. MAD uses the deviation from the median, which is less likely to be skewed by outlier values.

$$\alpha = \widetilde{MSE}_{benign\_val} + 5 \times MAD(MSE_{benign\_val}) \quad (5)$$

Algorithm 1: AE-based Anomaly Detection

BEGIN
PHASE 1: Flow extraction & Preprocessing: #
INPUT: $pkts$: raw packets, TW: Time window (second)
Extract flow from $pkts$ according to the TW
Calculate l features vectors (see table I for features list)
####
PHASE 2: Training & Testing the model #
INPUT: Benign dataset X, Malicious dataset $x^{(i)}$ $i \in \{1, ..., N\}$, threshold α
OUTPUT: RE (Reconstruction Error) $\|x - \hat{x}\|$
$\phi, \theta \leftarrow$ train the AE on the benign dataset X
for $i \in \{1, ..., N\}$ **do**
  $RE(i) = \left\|x^{(i)} - g_\theta(f_\phi(x^{(i)}))\right\|$
  if $RE(i) > \alpha$ **then**
    $x^{(i)}$ is a malicious flow
  else
    $x^{(i)}$ is a benign flow
  end if
end for
END

C. MEC-enabled federated training process

First, the MEC server initializes the learning parameters of the shared model in terms of neural network configuration (number of layers, number of neurons, activation functions, etc.), batch size, learning rate, number of epochs, etc. It then shares such parameters with the CAV collaborators. Each CAV computes local updates on top of both the AE model and its local data. Once done, each CAV sends its local model's parameters (weights) to the MEC server. The latter aggregates the received local models to generate a global learning model, before sending it back to the involved CAVs, in order to initiate a new training round.

In our study, the federated learning problem across multiple CAVs is formulated as a federated optimization problem and resolved using the FedAvg algorithm [16]. Indeed, using its local data, each CAV calculates the average gradient on top of the model w for a corresponding training round r. Thereafter, each CAV performs a local gradient descent on the currently used model with its own data. On the other hand, the MEC server aggregates these local updates and transfers back the global model to the CAV collaborators. This process is repeated during a number of rounds, defined initially by the MEC server. **Algorithm 2** illustrates the main steps performed by both the central MEC server and each participating CAV (Client).

Algorithm 2: Federated Averaging Algorithm

BEGIN
Variables: *K*: index of clients, *B*: local batch size, *E*: number of local epochs, η : learning rate
ClientUpdate: #
$\beta \leftarrow$ (split P_k into batch of size B)
for *each local epoch* $i \in \{1, ..., E\}$ **do**
  for *batch* $b \in B$ **do**
    $w \leftarrow w - \eta \bigtriangledown l(w; b)$
  end for
  return $w$to server
end for
Server executes:
Initialize w_0
for *each round* $t \in \{1, ..., N\}$ **do**
  $m \leftarrow$ max(C. K, 1)
  $S_t \leftarrow$ (random set of m clients)
  for *each client* $k \in S_t$ *in parallel* **do**
    $w_{t+1}^k \leftarrow ClientUpdate(k, w_t)$
    $w_{t+1} \leftarrow \sum_{k=1}^{k} \frac{n_k}{n} w_t^k + 1$
  end for
end for
END

IV. Performance Evaluation

In this section, we first briefly describe the dataset [17] used in this research. Then, we present and discuss in detail the detection performances of the proposed system. Finally, we compare our approach with supervised and centralized approaches.

A. Dataset preprocessing & features enginnering

To the best of our knowledge, VDoS [17] is the only publicly available dataset that includes benign and malicious traffic generated based on a realistic testbed. The network traffic was gathered in three different settings: urban, rural, and highway. The experimental environment included two vehicles, 3 physical machines, 4 virtual machines, 2 access points, a 4G modem, and two Cisco antennas. Common user applications such as Google Maps, YouTube, social networks, and other real-time applications (video/audio calls) have been run to generate benign network traffic. To generate malicious traffic, three Kali-Linux machines run three scenarios of DoS attack: UDP Flood, SYN Flood, and Slowloris packets alternately. In this research, we do not consider the third scenario, because we believe it is quite unusual that a CAV may hosts a web server. For further details about the testbed and the dataset generation please refer to [17].

To extract flows and calculate features from raw traffic (PCAP files), we developed some scripts using Tranalyzer flow traffic exporter [18]. We tried several time windows (TW) to sample the network traffic. Having achieved similar performance, and taking into account that a smaller TW allows faster detection, we fixed the TW to 1s. Table II presents

TABLE II: Dataset samples distribution

	Highway	Rural	Urban
Normal	48303	32303	65742
SynFlood	65790	65790	67043
UdpFlood	65790	65790	65790

the number of benign and malicious flows extracted for each type of environment. Pearson correlation filter is then used to discard highly correlated features ($> 95\%$).

B. Experimental results

We trained and tested the proposed system in the Google Colab cloud environment. We used the Pytorch package to implement the local and federated learning models. We implemented a deep auto-encoder with three hidden layers (50% dimension decrease from one layer to another). The binary Mean Squared Error (MSE) loss function was used. Table III illustrates the hyperparameter setup used for local and federated learning. To evaluate the detection performance of our solution, in addition to the false positive rate (FPR), we considered the following metrics :

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{6}$$

$$\text{F1-Score} = \frac{2TP}{2TP + FP + FN} \tag{7}$$

Overall, the proposed solution presented a high detection rate with a low false positive rate. The detection model was able to discriminate between the reference benign traffic profile and malicious network traffic. Table IV shows how the average accuracy, F1-score, and false positive rate may change by varying the number of participating CAVs. Using just three CAVs allowed for nearly the same detection performances while cutting the training time by about 30%. We can see from figure 2, that there is no further improvement in terms of loss function score beyond the 11^{th} round. This demonstrates that the detection model did not require a large number of communication rounds to converge.

As can be observed from figure 3, the proposed system performed better against the SynFlood attack. This can be explained by the relatively large number of TCP-related features (compared to UDP) included in the features vector. Although 99.99% F1-Score and 0.01% FPR rate have been achieved using supervised learning, specifically, decision tree in [17], our system shows comparable detection performances using only benign traffic. To get a good idea of how well the proposed model worked, we compared it to a centralized model that was trained with the whole dataset. The results comparison depicted in figure 4 shows that the federated model yielded remarkably similar detection performances as the centralized model.

V. Conclusion

New attack vectors have emerged from the integration of V2X communication in the 5G ecosystem, which may lead to hazardous situations for road users. Most of the existing IDSs in 5G-V2X are either unable to detect emerging zero-day attacks because they rely on supervised learning, or do not meet privacy requirements due to data collection required for centralized learning. To tackle these limits, we proposed in this paper a new IDS based on a deep auto-encoder model, which leverages the predictability of benign network traffic to detect attacks. Relying on federated training orchestrated by the MEC server, the proposed IDS did not require any data collection or labeling. Through in-depth experiments on a recent dataset, we have shown that the proposed IDS provides high performance even with few communication rounds and a short TW sampling, which allows fast training and low detection delay. In future work, we plan to evaluate the proposed IDS on other non-Identical Independent Distribution (non-IID) datasets including more sophisticated and recent attacks.

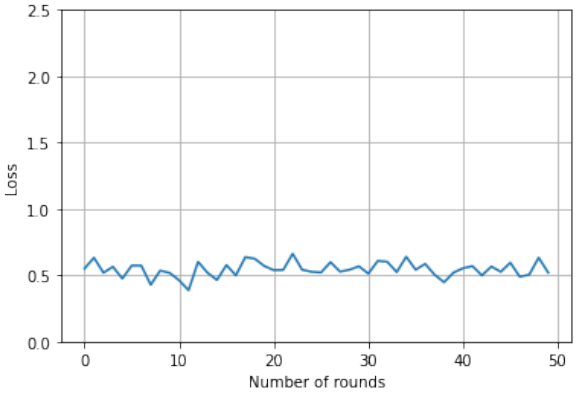

Fig. 2: Loss Vs Nb. rounds

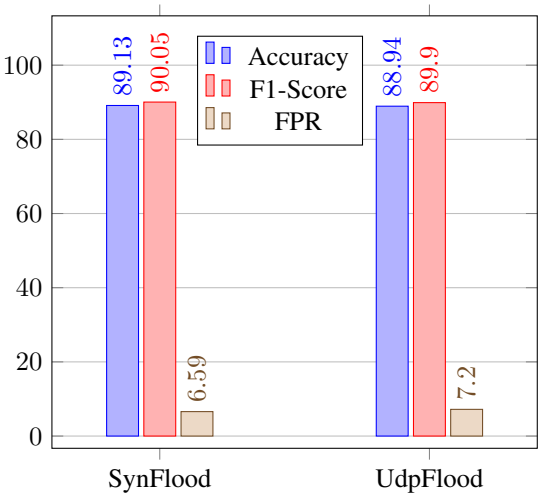

Fig. 3: Predictive performances per attack scenario

TABLE III: Hyperparameters values

Hyperparameter	Value	Description
lr	0.012	Learning rate for the deep AE
Nb_clients	10	Number of clients
Nb_selected	3-6	Number of clients we choose for train
Batch_size	128	Defines the dataset size in each training iteration
R_samp_sz	1000	Choose some data from the train set to retrain the data from trained model
Nb_rounds	20	Total number of communication rounds for the global model to train
Epochs	15	For train client model
Nb_retrain_epochs	5	Total number for retrian the global server after receiving the model weights
Nb_local_epochs	50	Only for the local deep AE training

TABLE IV: Evaluation of detection performances

Nb. Clients	Accuracy	F1-Score	FPR	Time (mn)
10	87.94%	91.21%	6.95%	14.80
8	87.94%	91.22%	6.98%	11.53
6	87.95%	91.23%	7.06%	9.32
3	87.94%	91.21%	6.92%	4.43

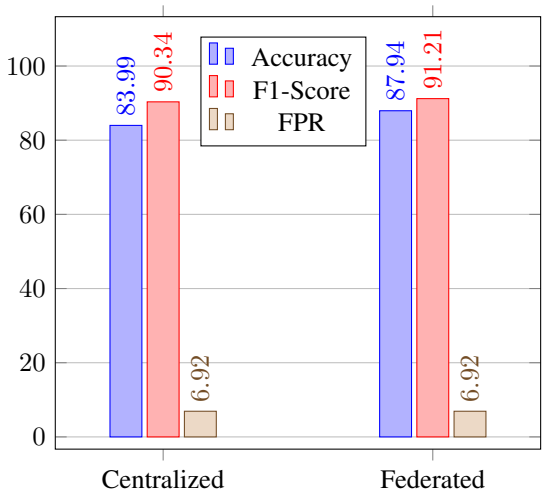

Fig. 4: Performances comparison between centralized and federated learning

ACKNOWLEDGMENT

This work was supported by the 5G-INSIGHT bilateral project (ID: 14891397) / (ANR-20-CE25-0015-16), funded by the Luxembourg National Research Fund (FNR), and by the French National Research Agency (ANR).

REFERENCES

[1] Ijaz Ahmad, Shahriar Shahabuddin, Tanesh Kumar, Jude Okwuibe, Andrei Gurtov, and Mika Ylianttila. Security for 5g and beyond. *IEEE Communications Surveys & Tutorials*, 21(4):3682–3722, 2019.

[2] Mario H. Castañeda Garcia, Alejandro Molina-Galan, Mate Boban, Javier Gozalvez, Baldomero Coll-Perales, Taylan Şahin, and Apostolos Kousaridas. A Tutorial on 5G NR V2X Communications. *IEEE Communications Surveys Tutorials*, 23(3):1972–2026, 2021.

[3] 3GPP TS 23.287. Architecture enhancements for 5G System (5GS) to support Vehicle-to-Everything (V2X) services, Jul 2020.

[4] Noble Arden Elorm Kuadey, Gerald Tietaa Maale, Thomas Kwantwi, Guolin Sun, and Guisong Liu. DeepSecure: Detection of Distributed Denial of Service Attacks on 5G Network Slicing-Deep Learning Approach. *IEEE Wireless Communications Letters*, 2021.

[5] Abdelwahab Boualouache and Thomas Engel. A Survey on Machine Learning-based Misbehavior Detection Systems for 5G and Beyond Vehicular Networks. *arXiv preprint arXiv:2201.10500*, 2022.

[6] Rabah Rahal, Abdelaziz Amara Korba, Nacira Ghoualmi-Zine, Yacine Challal, and Mohamed Yacine Ghamri-Doudane. Antibotv: A multilevel behaviour-based framework for botnets detection in vehicular networks. *Journal of Network and Systems Management*, 30(1):1–40, 2022.

[7] Wided Hammedi, Bouziane Brik, and Sidi Mohammed Senouci. Federated deep learning-based framework to avoid collisions between inland ships. In *2022 International Wireless Communications and Mobile Computing (IWCMC)*, pages 967–972, 2022.

[8] Naman Negi, Ons Jelassi, Hakima Chaouchi, and Stephan Clemençon. Distributed online Data Anomaly Detection for connected vehicles. In *2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, pages 494–500. IEEE, 2020.

[9] Jiangang Shu, Lei Zhou, Weizhe Zhang, Xiaojiang Du, and Mohsen Guizani. Collaborative intrusion detection for VANETs: a deep learning-based distributed SDN approach. *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[10] Tao Zhang and Quanyan Zhu. Distributed privacy-preserving collaborative intrusion detection systems for VANETs. *IEEE Transactions on Signal and Information Processing over Networks*, 4(1):148–161, 2018.

[11] Aashma Uprety, Danda B Rawat, and Jiang Li. Privacy Preserving Misbehavior Detection in IoV using Federated Machine Learning. In *2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC)*, pages 1–6. IEEE, 2021.

[12] Abdelwahab Boualouache and Thomas Engel. Federated learning-based scheme for detecting passive mobile attackers in 5G vehicular edge computing. *Annals of Telecommunications*, 77(3):201–220, 2022.

[13] Hong Liu, Shuaipeng Zhang, Pengfei Zhang, Xinqiang Zhou, Xuebin Shao, Geguang Pu, and Yan Zhang. Blockchain and Federated Learning for Collaborative Intrusion Detection in Vehicular Edge Computing. *IEEE Transactions on Vehicular Technology*, 2021.

[14] Amal Hbaieb, Samiha Ayed, and Lamia Chaari. Federated learning based IDS approach for the IoV. In *Proceedings of the 17th International Conference on Availability, Reliability and Security*, pages 1–6, 2022.

[15] Geoffrey E Hinton and Richard Zemel. Autoencoders, minimum description length and helmholtz free energy. *Advances in neural information processing systems*, 6, 1993.

[16] H. B. McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017.

[17] Rabah Rahal, Abdelaziz Amara Korba, and Nacira Ghoualmi-Zine. Towards the development of realistic dos dataset for intelligent transportation systems. *Wireless Personal Communications*, 115(2):1415–1444, 2020.

[18] S. Burschka and B. Dupasquier. Tranalyzer: Versatile high performance network traffic analyser. In *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–8, 2016.