Edge Computing-enabled Intrusion Detection for C-V2X Networks using Federated Learning

Aymene Selamnia*, Bouziane Brik*, Sidi Mohammed Senouci*, Abdelwahab Boualouache † and Shajjad Hossain*
* *DRIVE Laboratory* EA1859, Univ. Bourgogne Franche Comté, F58000 Nevers, France
† FSTM, Faculty of Science, Technology and Medicine, University of Luxembourg, Luxembourg
{aymene.selamnia, bouziane.brik, sidi-mohammed.senouci, Shajjad_Hossain}@u-bourgogne.fr
abdelwahab.boualouache@uni.lu

***Abstract*—Intrusion detection systems (IDS) have already demonstrated their effectiveness in detecting various attacks in cellular vehicle-to-everything (C-V2X) networks, especially when using machine learning (ML) techniques. However, it has been shown that generating ML-based models in a centralized way consumes a massive quantity of network resources, such as CPU/memory and bandwidth, which may represent a critical issue in such networks. To avoid this problem, the new concept of Federated Learning (FL) emerged to build ML-based models in a distributed and collaborative way. In such an approach, the set of nodes, e.g., vehicles or gNodeB, collaborate to create a global ML model trained across these multiple decentralized nodes, each one with its respective data samples that are not shared with any other nodes. In this way, FL enables, on the one hand, data privacy since sharing data with a central location is not always feasible and, on the other hand, network overhead reduction. This paper designs a new IDS for C-V2X networks based on FL. It leverages edge computing to not only build a prediction model in a distributed way but also to enable low-latency intrusion detection. Moreover, we build our FL-based IDS on top of the well-known CIC-IDS2018 dataset, which includes the main network attacks. Noting that, we first perform feature engineering on the dataset using the ANOVA method to consider only the most informative features. Simulation results show the efficiency of our system compared to the existing solutions in terms of attack detection accuracy while reducing network resource consumption.**

***Index Terms*—C-V2X, Intrusion detection system, Edge computing, Federated deep learning.**

I. Introduction

Emerging 5G networks have enabled lower latency, higher capacity, and increased bandwidth compared to 4G networks [1]. In vehicular context, the 5G technology boosted the existing services of both Intelligent Transport Systems (ITS), and Advanced Driver-Assistance Systems (ADAS) [2]. As an application of ITS, a vehicle-to-everything (V2X) network provides a great communication medium between vehicles and their surroundings (infrastructure, pedestrians, other vehicles, etc.). However, vehicular networks are characterized by a high mobility and hence a very dynamic topology and diversity of communication, which make them an easy target for various security attacks. Indeed, providing an efficient security mechanism is mandatory due to secure critical data exchanged in the network, in addition to the damage that can cause such attacks [3]. Attacks on vehicular networks can be classified into two main classes: First, the attacks targeting the intra-vehicle network, which aim to damage the internal network of the vehicles and hence disrupt the primary operations of ADAS; Second, the attacks targeting the inter-vehicle network, which aim to interrupt the connection of the vehicle with its environment, and hence disrupt the functioning of ITS systems. The latter class is critical due to the scale of the damage, which can vandalize the entire network. In this context, Intrusion Detection Systems (IDS) have proved their effectiveness in both detecting and mitigating susceptible attacks, especially when leveraging Machine Learning (ML) techniques and the huge traffic generated by C-V2X networks [4, 5].

In the literature, IDSs can be classified based on the system architecture into three types [4]: (i) Centralized, where a central node makes the detection, (ii) Distributed, where multiple nodes collaborate to perform the detection, and (iii) Hybrid, representing the combination of the two first types. Many centralized IDS solutions have been proposed [5]. However, the centralized IDS might become a system bottleneck, and it is also susceptible to a single point of failure. Distributed IDSs have thus emerged to mitigate the aforementioned scalability and reliability issues. Leveraging on edge computing, distributed IDSs share workloads across multiple components. This allows not only high scalability and reliability, but also fast processing and quick response time [6]. Besides, the IDSs can also be classified based on the detection method into misuse detection and anomaly detection. Misuse detection methods detect attacks based on the known attack signatures; they effectively detect known attacks with low errors. However, they cannot detect newly emerged attacks that do not have similar properties to known attacks. In contrast, the anomaly detection methods are based on the hypothesis that the attacker's behavior differs from that of a normal user. They classify network traffic as an attack if its characteristics are far from normal traffic patterns. However, this latter type requires data since it is based on ML techniques. The data used in this system plays a decisive role in the results obtained by the system. Hence, we value the data selection by studying different datasets and comparing them to choose the adequate dataset for our problem.

In this paper, we design a new distributed low latency

intrusion detection system for vehicle-to-everything networks. We leverage federated deep learning to train a prediction model in a federated way [7]. We first train several distributed models, with the same model architecture, on the MEC (Multi-Access Edge Computing) level using data collected from its covered network, then aggregate them to a global model in the cloud [8]. Finally, the aggregated model is sent again to the network edges to be used in predicting attacks.

The rest of this paper is organized as follows. Section II describes some related works. Section II-B presents and compares relevant datasets according to various criteria. Section III details the main steps in the design and the implementation of our distributed Intrusion Detection scheme (DID). Section IV describes the implementation settings and discusses the obtained results. Finally, Section V concludes the paper.

II. Related works

Intrusion detection systems proved their efficiency to secure vehicular networks, as they can secure the system from both internal and external attacks with high accuracy, especially when combined with machine learning techniques. These make it possible to exploit the huge amount of data in training accurate models with less time complexity. In this section, on one hand, we review existing ML-based IDS schemes that were designed for C-V2X. On the other hand, we also study existing datasets in order to select the suitable dataset for our case.

A. Machine/Deep Learning-based IDS

The authors of [9] proposed a collaborative IDS using the Generative Adversarial Network (GAN), by collecting the information flow from vehicles and RSUs, in the subnetwork of each distributed SDN (Sotware Defined Network) controller. Then, all the distributed SDN controllers and the cloud server train a global model, deployed at the SDN controllers to monitor the network flow independently. The training of this latter uses BiGANs, generative models that can learn the mapping from simple latent distributions to arbitrarily complex data distributions and generate realistic data [10]. When the IDS detects abnormal behavior, the system will manage the flow transmission and deal with the attacks using a flow of rules. This mitigation mechanism imposes rules to block abnormal flows.

The authors of [11] proposed an intrusion detection framework for vehicular networks, by designing a one-hop cluster-based architecture. The framework selects a cluster head through periodic exchange of status messages, including trust values of the neighbors, to choose the vehicle with the higher utility function. The framework includes two intrusion detection models: (1) a global model carried by the cluster head; its mission is to evaluate the trust level of monitored vehicles and to banish the malicious vehicles, using block list and store and forward technique in case no RSU is within the radio; and (2) a local model carried by the cluster members equipped with three techniques: (i) a rules-based detection technique (RBD) to detect an anomaly using a set of predefined rules; (ii) learning-based detection technique (LBD), an SVM algorithm that classifies the sources of the input into normal behaviors and anomalies; and (iii) a rules-based decision technique that inputs the results of the RBD and LBD to modify the neighbor's reputation.

Even the above works designed novel machine learning-based IDS for vehicular networks, either in centralized way [11] or distributed way [9]; however, the results of the solutions proposed in [9] remain questionable due to the usage of the KDD99 dataset for validation, which was generated in 1998. Indeed, this dataset was incredibly valuable at the time of release, but it lost most of its relevance through aging and the change in the network architecture [12]. In addition, the solution designed in [11] cannot deal with some critical attacks, such as Sybil attacks, that can cause severe damage to the network due to the usage of trust-based systems.

B. Available Datasets

Due to the importance of the training data in anomaly-detection-based IDS, we dedicate this subsection to reviewing existing datasets related to IDSs, in order to select the suitable one for our IDS.

There are many datasets created over the last few years to deal with intrusion detection. Researchers of the University of California processed the Tcpdump of a prior dataset called DARPA. It was created by MIT Linkon laboratory in 1998 to create another dataset called KDD-Cup 1999. However, this latter consists of a lot of redundant and duplicate data samples. Researchers in [13] tried to cleaning it and creating a better version called NSL-KDD in 2009.
CAIDA is another dataset specific to DDOS attacks developed by the Center of Applied Internet Data Analysis in 2007. ISCX IDS 2012 was created by observing the alpha and beta profile of the network packets [14].
AFDA dataset contains features that show attack patterns and system call traces [15]. CIC-IDS-2017 and CSE-CIC-IDS-2018 datasets were generated by the Canadian Institute of Cyber Security in 2017 and 2018, respectively. They contain 80 features compared to prior datasets. It lists a new range of attacks generated from real network traffic features, such as Distributed Denial of Service, Denial of Service, brute force, XSS, SQL Injection, Botnet, Web attack, and Infiltration [16].

The authors of [17] compared the most used data sets in intrusion detection. In Table I, we extend their comparison to cover recent datasets as well as more comparison parameters. We give high importance to the generation environment and the year of generation, to stay as close as possible to real-world scenarios and the coverage of the denial of service attacks in both centralized and distributed ways, considering their devastating impact. Based on this comparison, we chose the recent CIC-IDS2018 dataset since it contains not only various features related network communication (80 features), but also it includes the main attacks that can target V2X networks, including DoS, DDoS, Botnet, brute force, etc.

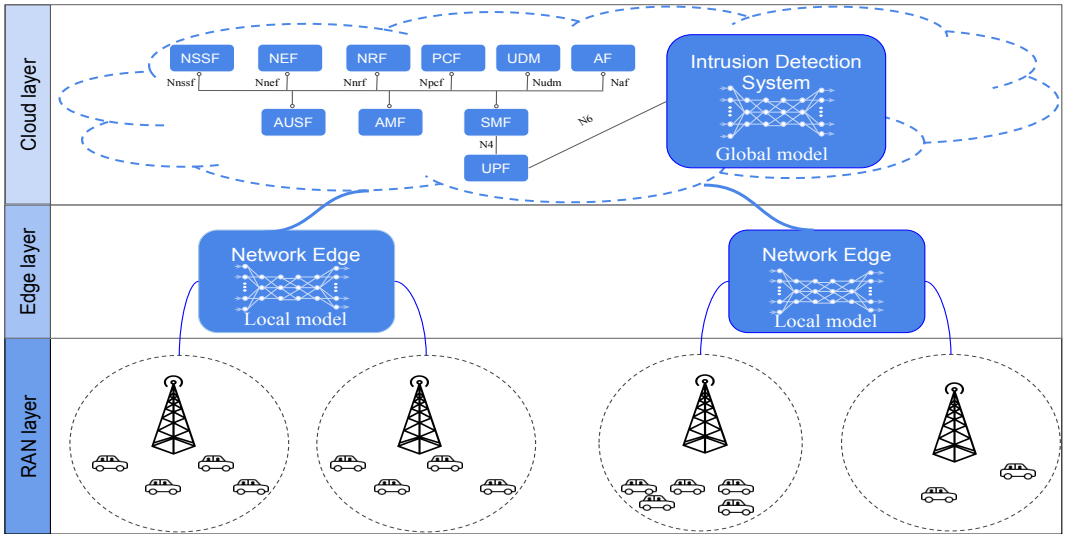

Fig. 1. Architecture overview

TABLE I
DATASETS COMPARAISON

	Realistic Traffic	Labeled data	0-day Attacks	Full Packet capture	DOS	DDOS	# features	# records	Files format	generation environment	Year
KDD-Cup	YES	YES	NO	YES	YES	NO	41	5M	CSV	IEEE 802.3	1999
Caida	YES	NO	NO	NO	NO	YES	-	-	PCAP	IEEE 802.3	2007
NSL-Kdd	YES	YES	NO	YES	YES	NO	41	5M	CSV	IEEE 802.3	2009
ISCX - 2012	YES	NO	NO	YES	YES	YES	-	-	PCAP/XML	IEEE 802.3	2012
ADFA-LD	YES	YES	YES	YES	NO	NO	26	90M	CSV	IEEE 802.3	2014
UNSW-NB 15	NO	YES	NO	YES	YES	NO	49	257K	CSV	IXIA traffic generator	2015
CIC-IDS 2018	YES	YES	YES	YES	YES	YES	79	15M	PCAP/CSV	IEEE 802.3	2018
AWID	YES	YES	NO	YES	YES	NO	155	210M	CSV	IEEE 802.11	2021

III. DID: DISTRIBUTED INTRUSION DETECTION SYSTEM FOR C-V2X

In this section, we describe our distributed intrusion detection system for C-V2X, leveraging federated learning. We start first presenting the selected dataset, followed by the pre-processing as well as feature selection steps. Then, we present the general neural network architecture. Finally, we outline the tuning study to determine the optimal model parameters for the best performance results. Before we proceed, we provide an overview about the IDS system architecture.

A. IDS System Architecture Overview

Fig. 1 shows the general architecture of our solution. It consists of three main layers (cellular network layers): (i) the radio access network (RAN) where the main task of this layer is data collection; the packets exchanged in this layer will be forwarded to the next layer; (ii) the edge Layer which is in charge of extracting the essential features, to train the local models from the data received from the RAN layer using CIC-FlowMeter v3. (iii) the Core Layer where its main task is to aggregate the local models, received from the edge layers, into a global model. This global model will be distributed to the clients (network edges) to be used in predicting attacks from the flow received from the RAN layer.

B. Federated Learning-based Prediction Model

1) Dataset Overview: CIC-IDS2018 results from a collaborative project between the Communications Security Establishment (CSE) and The Canadian Institute for Cybersecurity (CIC). It includes seven different attack scenarios, namely Brute-force, Heartbleed, Botnet, DoS, DDoS, Web attacks, and infiltration of the network. The attacking infrastructure includes 50 machines, and the victim organization has five departments, including 420 PCs and 30 servers. This dataset includes the network traffic and logs files of each machine from the victim side, along with 80 network traffic features extracted from captured traffic using CICFlowMeter-V3[18].

2) Data Pre-processing: Data pre-processing is the first step of the machine learning process. The dataset is the output of CICFlowMeter-V3, which contains 80 features (columns) and around 15 million records (rows). It contains irrelevant data to our problem, and several missing values or infinite values in some records. We addressed those problems by first deleting the missed values records, and we omitted data that we judged irrelevant to our problem. Most of the omitted data was related to the generation environment (Source IP address, port, etc.). We also formatted the data in a suitable format, by encoding categorical data using one-hot-encoding. One more step in formatting our data is to split it into training and test subsets; we opted for 70% for the training and 30% for the test.

3) Features Selection: Eliminating useless (irrelevant and/or redundant) features enhances the accuracy of classification, while speeding up the training step [19]. Thus, we determine the features with the higher correlation with the labels. After that, we chose to work with 30 features as the correlation seems to drop after the thirtieth feature. This will enhance the accuracy and reduce the calculation time to the third.

4) Deep Learning Architecture: The deep learning model consists of five layers: one input layer with 30 neurons that correspond to the input features, passed to four hidden layers of 60, 120, 60, and 30 neurons, respectively, with rectified linear activation function (RELU) passed to 12 neurons of the output layer, that correspond to the 12 attacks covered by our solution, with softmax activation. We opted for cross-entropy loss and ADAM optimizer for neural weights' updates.

5) Federated Training Process: Federated learning is a category of collaborative learning framework where several participants collaborate in training a global model. During the training step, the central node trains a global model in a distributed manner and then shares the model with all participants. First, the central node initializes a global model and specifies its hyper-parameters, such as learning rate and the number of epochs. Then, it shares them with all the participants to train a local version of it, using the local data of each participant. After that, the central node collects the model weights from all participants and aggregates them to one global model, using an aggregation function (Federated Averaging FedAvg [20] in our case). Finally, it updates the weights of the global model with the aggregated weights.

IV. Experimental Study

A. Implementation Settings

To get the best performance, we studied various configurations of the model hyper-parameters. In the following, we detail how we proceeded to choose the model parameters, such as learning rate, batch size, and epochs number.

1) Learning Rate: To find the appropriate learning rate for the proposed model, we used an open-source implementation of the technique proposed in [21], that varies the learning rate and plots the changes in the loss. Fig. 3 [A]) shows that from 10^{-5} to 10^{-4}, the loss is almost fixed; the learning rate is too low for the network to learn anything. From about 10^{-3}, the loss starts to decrease; this is the lowest learning rate where our network can learn. By the time we reach 10^{-1}, our model is learning very quickly. At just over 10^{-1}, we see an increase in loss. Finally, at 0.5, our loss has exploded; the learning rate is far too high for our model to learn. Looking at this graph, we can determine our learning rate's lower and upper bounds. Specifically, the lower bound equals 10^{-3}, while upper bound = 10^{-1}.

2) Batch size: To determine the suitable batch size, we evaluate the performance of our model on different batch sizes. Fig. 3 [B]) shows the accuracy, precision, and loss for different batch sizes. We can see that we choose between the values of 64, 128, and 254, since they have the same performance. To balance the computational cost and escape the local minimums, we decide to work with 64.

3) Number of Epochs: To choose the number of epochs, we vary the number of epochs and study the changes in the loss. Fig. 3 [C]) shows that between 1 and 2 epochs, the loss continues to decrease. Nevertheless, from the 3^{rd} epoch, our loss function starts to stabilize, which means that the suitable number of epochs is between 3 and 6 epochs. We have chosen to work with 3 epochs, because our model is more efficient compared to other numbers of epochs.

B. Performance Evaluation in Terms of Prediction

Table II summarizes the results obtained on the test set. The results show that our solution can predict 95% of the traffic in the test set correctly. More specifically, 92% precision means that if our model predicts an attack, it is 92% correct. In addition, a recall of 0.94 means it correctly detects 94% of the attacks.

To understand the behavior of our system with the different attacks, we also computed the confusion matrix. The role of this matrix is to present the distribution of our prediction broken down by classes. This allows us to see where our model is confused. Table III shows the predictions of our system, the diagonal represents the attacks correctly predicted by our system, and the other cells show the number of false predictions and the label assigned to them. We notice that our system confuses attacks with a small number of instances (brute force attacks, SQL injection, and infiltration attacks), with other attacks or even with benign traffic, but successfully detects the rest of attacks.

C. Performance Evaluation in Terms of Network Overhead

When generating a learning model in federated way, network overhead represents one of the important performance indicators. Through our distributed solution, with the usage of federated learning, we were able to reduce 95% of network traffic, as the following calculations demonstrate. Indeed, for the federated learning, the network overhead corresponds to the number of sent packets, to exchange the local models and global model. Thus, *Number_of_packets = (Client_packets X departments_number X rounds) + initialization_packet*, which equal to *Number of packets = (5X5X5)+ 1 = 126 packets*. Considering a maximum packet size of 1500 bytes, the total size of exchanged packets is *number of packets * packet size = 126 * 1.5 MB = 189 MB*. On the other hand, the Dataset size (after features selection) is 5,625 Go with *5.625 - 0.189 = 5.436 Go.* of data are kept at the clients. Therefore, using federated learning, we were able to avoid exchanging 5.436 Go of data network, which represent 95% of network traffic.

D. Comparison

To better understand the behavior of our solution against attacks, we have chosen three attacks to see how our solution will react. We have chosen two attacks with many instances: DoS and DDoS, and one with a reduced number of instances, which is the infiltration attacks. Figs 3 represent the predictions of our system in 40 instant simulations. We compare

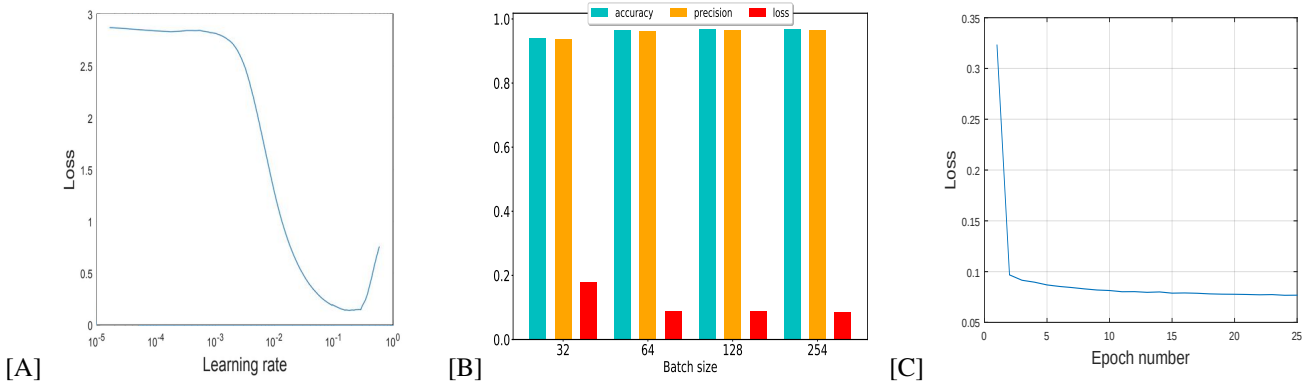

Fig. 2. Implementation settings, [A] Learning rate, [B] Batch size, [C] Number of epochs

TABLE II
MODEL PERFORMANCE

	Accuracy	Precision	Recall	F1-score	Detection rate	False alarm rate	AUC
Results of centrelized model	0.96451	0.9596	0.9321	0.9721	0.6041	0.0087	0.9987
Results of federated learning model(DID)	0.95813	0.9247	0.9448	0.9644	0.5806	0.0094	0.9985

TABLE III
CONFUSION MATRIX

	Benign	Bot	Brute force-web	Brute force XSS	DDoS attack HOIC	DDoS attack LOIC-UDP	DDoS attack GoldenEye	DoS attack Slowloris	FTP Brute force	Infiltration	SQL Injection
Benign	0.87	0.02	0	0	0	0	0.02	0	0	0.09	0
Bot	0	0.99	0	0	0	0	0	0.0003	0	0.0007	0
Brute force-web	0.61	0	0	0	0.02	0	0	0.11	0	0.24	0
Brute force XSS	0.27	0	0	0	0	0	0	0.33	0	0.38	0
DDoS attack HOIC	0.001	0	0	0	0.99	0	0	0	0	0	0
DDoS attack LOIC-UDP	0	0.024	0	0	0	0.97	0	0	0	0.006	0
DDoS attack GoldenEye	0.001	0	0	0	0	0	0.999	0	0	0	0
DoS attack Slowloris	0.63	0	0	0	0	0	0.34	0	0	0.01	0
FTP Brute force	0	0	0	0	0	0	0	0	1	0	0
Infiltration	0.35	0.003	0	0	0	0.001	0	0.003	0.001	0.6411	0
SQL Injection	0.38	0.12	0	0	0.3	0	0	0	0	0.2	0

the results of the three models with the real traffic. For both DoS and DDoS attacks, the graphs are overlapped, proving that our solution and the two other models were able to detect the attacks. On the other hand, the graphs of the infiltration attacks do not overlap with the real network graph, which means that we miss-classify some traffic due to the size of the training data for this type of attack. These results show that our federated learning-based IDS can effectively detect DoS and DDoS attacks, while reducing the network overhead, thanks to the federated learning technique.

V. CONCLUSION AND PERSPECTIVES

In this work, we propose a novel intrusion detection system for C-V2X networks, capable of detecting many attacks that may severely impact the normal operation of this type of networks, such as DoS, DDoS, and botnets. All while considering the response time, thanks to federated learning and edge computing, we bypass a significant data exchange between the core network and the vehicle. Future work could be testing the effectiveness of our solution in real scenarios and increasing the accuracy of the deep learning model by building a more diverse and complete data set. Exploiting the concept of transfer learning to cover more attacks could be a good addition, especially when it comes to attacks on network slices. We can also extend our system to detect 0-day attacks by adding a complementary unsupervised learning model.

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] B. Brik, K. Boutiba, and A. Ksentini, "Deep learning for b5g open radio access network: Evolution, survey, case studies, and challenges," *IEEE Open Journal of the Communications Society*, vol. 3, pp. 228–250, 2022.

[2] B. Brik, N. Lagraa, Y. Ghamri-Doudane, and A. Lakas, "Finding the most adequate public bus in vehicular clouds," in *2016 International Conference on Wireless Networks and Mobile Communications (WINCOM)*, 2016, pp. 67–74.

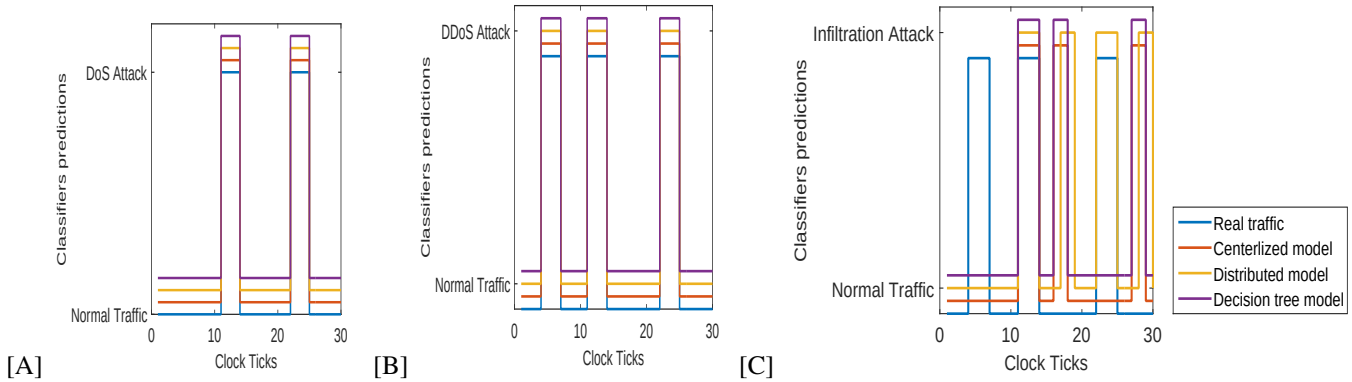

Fig. 3. Prediction compared to real traffic, [A] DoS attack, [B] DDoS attack, [C] Infiltration

[3] T. Eddine Toufik Djaidja, B. Brik, S. Mohammed Senouci, and Y. Ghamri-Doudane, "Adaptive resource reservation to survive against adversarial resource selection jamming attacks in 5g nr-v2x distributed mode 2," in *ICC 2022 - IEEE International Conference on Communications*, 2022, pp. 3406–3411.

[4] F. Sabahi and A. Movaghar, "Intrusion detection: A survey," in *2008 Third International Conference on Systems and Networks Communications*, 2008, pp. 23–26.

[5] H. Bangui and B. Buhnova, "Recent advances in machine-learning driven intrusion detection in transportation: Survey," *Procedia Computer Science*, vol. 184, pp. 877–886, 2021.

[6] Z. A. E. Houda, B. Brik, and L. Khoukhi, ""why should i trust your ids?": An explainable deep learning framework for intrusion detection systems in internet of things networks," *IEEE Open Journal of the Communications Society*, vol. 3, pp. 1164–1176, 2022.

[7] Z. Abou El Houda, B. Brik, A. Ksentini, L. Khoukhi, and M. Guizani, "When federated learning meets game theory: A cooperative framework to secure iiot applications on edge computing," *IEEE Transactions on Industrial Informatics*, pp. 1–1, 2022.

[8] B. Brik and A. Ksentini, "Toward optimal mec resource dimensioning for a vehicle collision avoidance system: A deep learning approach," *IEEE Network*, vol. 35, no. 3, pp. 74–80, 2021.

[9] J. Shu, L. Zhou, W. Zhang, X. Du, and M. Guizani, "Collaborative intrusion detection for vanets: A deep learning-based distributed sdn approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 4519–4530, 2021.

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Advances in Neural Information Processing Systems*, vol. 3, 06 2014.

[11] H. Sedjelmaci and S.-M. Senouci, "A new intrusion detection framework for vehicular networks," *2014 IEEE International Conference on Communications (ICC)*, pp. 538–543, 2014.

[12] G. Creech and J. Hu, "Generation of a new ids test dataset: Time to retire the kdd collection," in *2013 IEEE Wireless Communications and Networking Conference (WCNC)*, 2013, pp. 4487–4492.

[13] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the kdd cup 99 data set," in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, 2009, pp. 1–6.

[14] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *Comput. Secur.*, vol. 31, pp. 357–374, 2012.

[15] G. Creech and J. Hu, "Generation of a new ids test dataset: Time to retire the kdd collection," *2013 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 4487–4492, 2013.

[16] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *ICISSP*, 2018.

[17] M. Malowidzki, P. Berezi, ski, and M. Mazur, "Network intrusion detection : Half a kingdom for a good dataset," 2015.

[18] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *ICISSP*, 2018.

[19] S. Zaman and F. Karray, "Features selection for intrusion detection systems based on support vector machines," in *2009 6th IEEE Consumer Communications and Networking Conference*, 2009, pp. 1–8.

[20] T. Sun, D. Li, and B. Wang, "Decentralized federated averaging," *arXiv preprint arXiv:2104.11375*, 2021.

[21] L. N. Smith, "Cyclical learning rates for training neural networks," in *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2017, pp. 464–472.