A Lightweight 5G-V2X Intra-slice Intrusion Detection System Using Knowledge Distillation

Shajjad Hossain*, Abdelwahab Boualouache**, Bouziane Brik*, and Sidi-Mohammed Senouci*

*DRIVE Lab, University of Burgundy, Nevers, France, email: {firstname.lastname}@u-bourgogne.fr
**FSTM, University of Luxembourg, Luxembourg, email: {firstname.lastname}@uni.lu

***Abstract*—As the automotive industry grows, modern vehicles will be connected to 5G networks, creating a new Vehicular-to-Everything (V2X) ecosystem. Network Slicing (NS) supports this 5G-V2X ecosystem by enabling network operators to flexibly provide dedicated logical networks addressing use case specific-requirements on top of a shared physical infrastructure. Despite its benefits, NS is highly vulnerable to privacy and security threats, which can put Connected and Automated Vehicles (CAVs) in dangerous situations. Deep Learning-based Intrusion Detection Systems (DL-based IDSs) have been proposed as the first defense line to detect and report these attacks. However, current DL-based IDSs are processing and memory-consuming, increasing security costs and jeopardizing 5G-V2X acceptance. To this end, this paper proposes a lightweight intrusion detection scheme for 5G-V2X sliced networks. Our scheme leverages DL and Knowledge Distillation (KD) for training in the cloud and offloading knowledge to slice-tailored lightweight DL models running on CAVs. Our results show that our scheme provides an optimal trade-off between detection accuracy and security overhead. Specifically, it can reduce security overhead in computation and memory complexity to more than 50% while keeping almost the same performance as heavy DL-based IDSs.**

***Index Terms*—5G-V2X; Security; Deep learning; IDS; Knowledge Distillation; Network Slicing.**

I. Introduction

The Fifth Generation (5G) cellular network is offering a wide range of services to meet various needs in numerous industries, including augmented reality, smart homes and cities, industry 4.0/5.0, health care, and Intelligence Transport Systems (ITS) [1]. Particularly, for ITS, 5G brings a major revolution by integrating 5G-V2X communication technology into the next-generation cellular networks [2]. 5G-V2X has enabled innovative V2X use cases and applications such as vehicle platooning, and advanced remote driving taking advantage of data provided by other 5G verticals [3, 4]. More importantly, 5G-V2X has the potential to dramatically reduce traffic accidents and save lives, real-time surrounding awareness for Connected and Automated Vehicles (CAVs) [5]. Besides, 5G connectivity and access to 5G enabling technologies such as Network Function Virtualization (NFV), Software Defined Networking (SDN), Multi-access Edge Computing (MEC), and NS will make CAVs more connected than ever. In this context, CAVs may attach to several V2X network slices tailored to meet the requirements of V2X applications and establish independently isolated connections. Unfortunately, NS's benefits are underlying new cybersecurity challenges to CAVs. Specifically, V2X-NSs are vulnerable to several attack surfaces exposed by NS-enabling technologies (SDN and NFV)[6][7]. These attacks become even more complex and widespread and can be classified as intra-slice and inter-slice attacks. Intra-slice attacks can be launched inside a V2X network slice potentially targeting the application using that slice, while inter-slice attacks could prevent CAVs to switch between V2X network slices or prevent optimal slice management [8][9]. For thwarting these attacks, various deep learning IDSs have been proposed to efficiently secure V2X networks. These solutions have already proven their attack detection effectiveness while coping with large amounts of data [10]. However, Deep Learning models, on the other hand, tend to be quite large in order to increase accuracy and discover patterns from data. As a result, classic DL-based IDSs are computational and memory intensive, and are very prone to produce security overheads [11].

Deploying a DL-based IDS on CAV attached to one network slice might be acceptable. However, in a multiple-slice scenario, deploying a cumbersome DL-based IDS for each V2X network slice would be resource-heavy at the stake of decision-making services. In addition, using one DL-based IDS to detect attacks in all the running V2X network slices fails to meet the isolation and privacy preservation requirements of NS. Besides, since each V2X network slice comes with its specificity and uniqueness of data flowing through it, data distribution could be significantly different from one slice to another for the same type of attack. To address the previously-mentioned issues, deploying lightweight DL-based IDS is mandatory. To this end, our paper proposes a novel lightweight intrusion detection scheme for 5G-V2X sliced networks. Our scheme leverages DL and Knowledge Distillation (KD) for training in the cloud and offloading knowledge to slice-tailored lightweight models running on CAVs, mimicking the teacher-student learning process. Our results show that our scheme provides an optimal trade-off between detection rate and security overhead. Specifically, our scheme can reduce security overhead in computation and memory complexity to more than 50% while keeping almost the same performance as heavy DL-based IDSs.

The remainder of this paper is organized as follows. Section II describes the main related work on attack detection in 5G-V2X networks. Our system model and the intra-slices IDS is presented in Section III. Section IV discusses the

performance evaluation of the IDS on the VeReMi extension dataset. In order to evaluate the IDS, data generation and data pre-processing are also described in this section. Finally, Section V concludes the paper.

II. Related Work

In the past decade, various types of deep learning-based IDS have been designed to secure V2X networks. The authors in [12] built a DL-based IDS using two DL architectures: stacked Long Short-Term Memory (LSTM) and Convolutional Neural Networks (CNN)-LSTM. The previous work was extended in [13]. The authors proposed similar DL models and suggested deploying them as a detection engine in the MEC. The authors considered two detection methods: (1) Sequence Classification and (2) Sequence-image classification. They developed two models for sequence classification: (i) four stacked layers LSTM and (ii) CNN-LSTM. They also developed two models, CNN and Multi-Layer Perceptron (MLP), for sequence-image classification. The authors in [14] applied a stacked LSTM to identify suspicious vehicle behavior. They defined a multi-tier network and used it to deploy their model. The authors in [15] proposed a DL-based IDS to detect the same attacks considered in their previous work but based on an unsupervised learning approach. They trained solely on normal data using DL architectures similar to auto-encoders. Their model learns to reconstruct only normal data, thus when anomalous input is introduced, the reconstruction deviates significantly from the normal distribution. The same authors extended this work in [16] by considering further models for enhancing detection performance. The authors of [17] proposed a DL-based IDS deploying a set of DL-empowered security Virtual Network Functions attacks within V2X network slices. The authors in [18] used a hybrid collaborative approach based on rules and a Generative Adversarial Network (GAN) approach to secure MEC nodes. The authors in [19, 20] proposed a DL-based IDS for Distributed Denial of Service (DDoS) attacks in 5G networks.

The previous works have proposed IDSs based on complex DL architectures to provide high accuracy in detecting V2X attacks. However, such heavy DL-based IDSs might jeopardize the decision-making services of CAVs, especially in V2X network-sliced environments, as stated in the introduction. To address these issues, we propose a novel lightweight intrusion detection scheme for 5G-V2X sliced networks that takes advantage of DL and KD to provide an optimal trade-off between detection accuracy and security overhead.

III. Knowledge Distillation based Intra-slice V2X IDS

This section describes KD-based Intra-slice V2X IDS. We first describe the 5G-V2X network architecture targeted by our proposed IDS. Then, we give background on Knowledge Distillation and the teacher-student paradigm used in our scheme.

A. The targeted 5G network architecture

In Figure 1, the network architecture considered for the IDS deployment scheme is described. In this architecture, CAVs are connected to gNodeB by the Uu interface, and the $PC5$ sideline interface is used for V2V data transmission. In the V2X network context, each vehicle may be involved in multiple NSs based on the application requirements. As has already been discussed in the introduction, having a cumbersome model dedicated to each slice is unsuitable. In the proposed work, a lightweight IDS is proposed to detect attacks for each network slice at the application level. The DL models run on the vehicle's On-Board Unit (OBU), while the training is realized in the cloud before the deployment.

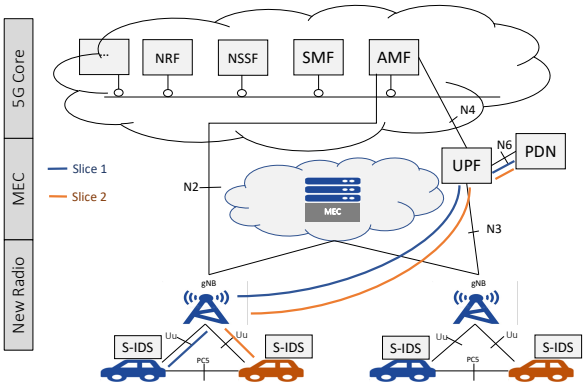

Fig. 1: 5G Network Architecture for IDS deployment and training.

B. Knowledge Distillation for V2X DL-based IDS

Knowledge Distillation (KD) is a particular method among other knowledge transfer between neural networks and DL model compression methods, that has been developed in recent years [21]. KD allows substituting a cumbersome model or an ensemble of models which is called a "Teacher" model by a lightweight model, which is called a "Student" model. The objective of KD is to reduce the complexity and computing resources for the DL models or to improve the performance of a model without adding complexity. In a generic case, KD can be modeled as follows:

Considering a dataset D with features and labels $(x.y) \in D$ and a model T. On the dataset D, the model T is trained using the desired objective function. This model can be as complex or have as many parameters as needed to best fit the dataset. When the training of the model T is done and it has reached the desired accuracy or other evaluation metrics, then another model S called the student model is constructed. The goal of this model is to be lightweight and to have lesser numbers of parameters while having similar performance.
As Hinton et al. [11] have already demonstrated, depending on the dataset, the dataset for training the student model can also be significantly reduced, requiring much less time and resources. Consequently, to train the student model, a smaller random subset D' is chosen from the initial training dataset

D. When choosing D', it is important to maintain in D' the different class distributions given in D. Otherwise, there will be an imbalanced dataset. After choosing the subset D', the student is trained with it and the distilled knowledge from the teacher model T. The distilling of knowledge from the teacher model to the student model is done by matching the output logits [1] of the teacher model by the student model. It comes down to reducing the error between the teacher's model output and the student's model output. The student model considers the teacher's model output as soft labels. Therefore this computes the gradient of the error and back propagates to update its weights.

$$L\_ts_{(t)} = \frac{1}{|D'| * t^2} * KL(S_{(x)}, T_{(x)}) \tag{1}$$

The loss function for matching logits can be written down as in Equation 1. In this paper, to calculate the difference between the teacher's and student model's outputs, Kullback–Leibler divergence (KL) is used. However, other distance-measuring functions can be also used; the only important part is to calculate the distance between two probability distributions. KL is similar to cross-entropy and measures the distance between the teacher and student's output. Contrary to cross-entropy, KL is more stable when the student's output matches the teacher's output hence producing better results. To apply KL or cross-entropy, before calculating the loss, both the student and teacher logits are transformed to probabilities by applying softmax.

Hinton et al showed in [11] that the student model can learn just by minimizing the loss function in Equation 1. However, they also demonstrated, in order to get better results, distilled knowledge can be combined with the true outputs from D' which are called hard labels. The loss for hard labels can be any function that measures the distance between two probability distributions such as cross-entropy. Finally, both of the loss functions for hard and soft labels can be combined and written as in Equation 2. In Equation 2, the parameter t is the temperature that divides the logits. Using a higher value for t produces a softer probability distribution over classes.

$$L\_ts_{(t,\alpha)} = \sum_{(x,y)\in D'} t^2 * \alpha * KL\left(S_{(x)}, T_{(x)}\right) + (1-\alpha) * Lc\left(S_{(x)}, y\right) \tag{2}$$

Where Lc is the cross entropy between the model S's output and the true outputs y of the dataset D'; and the parameter α allows to balance of the influence distilled knowledge from the teacher and the hard labels from the dataset D'. Notice the distillation loss function is multiplied by t^2. This is because in equation 1 the loss is divided by $1/t^2$, so it guarantees that even if the distillation temperature t is changed during the training phase, the relative contributions of the hard and soft targets will remain unchanged [11, 22]. Depending on the dataset and models used, multiple loss functions can be employed to match hard and soft labels and combine them as in Equation 2. Once the desired loss function and its parameters are specified, the student model is trained with it, and, after it has achieved the desired performance, we deploy the model.

[1]logits: raw values that come out from the last layer of a neural network.

IV. Intra-slices IDS Deployment on a V2X Application

This paper is supported by the 5G-INSIGHT bilateral project between France and Luxembourg [23]. Our experiments were done according to one of the use cases covered in the project. More precisely, the targeted use case is real-time traffic flow regulation. This use case would make use of the different onboard sensing technologies available in vehicles (Lidar, radar, GPS, etc.) to predict and control traffic. However, other complementing information derived from external sources such as traffic cameras, roadside units (RSU), and other sensors, would also be required. For the demonstration, only data from the onboard sensors are considered. In the following, the real-time traffic control application, used to evaluate the IDS, is introduced. Newt, VeReMi extension dataset generation is explained with data preprocessing. Following that, the IDS teacher's and student's models implementation and evaluation are detailled.

A. Application context and architecture for real-time traffic flow regulation

To implement this use case, one could imagine a variety of architectures. For our experiments, we consider a simple distributed architecture where an application is deployed on each vehicle's onboard unit (OBU) using a dedicated NS, and the traffic regulation is done collaboratively. One of the advantages of distributed nature of the application is that the traffic regulation will still work without any connection to 5G-NR or any cloud infrastructure. Only by using the PC5 interface, the vehicle can exchange the necessary information with each other. This type of distributed scenario could be particularly useful when the vehicle is temporarily disconnected from the 5G core network and other infrastructure. For this considered application, CAVs need to exchange at least information about their position, heading, and speed in order to take a proper collective decision. Before taking any decision, a CAV also has to verify the correctness of the information received by its neighboring vehicles. This is the responsibility of the proposed lightweight IDS system, which is constituted of DL models and can be deployed in each vehicle, making each vehicle responsible for its own security.

B. Dataset preprocessing & features enginnering

For the demonstration of the aforementioned scenario, the VeReMi Extension dataset [24] was used. VeReMi Extension is a publicly available dataset generated by using VEINS an open-source simulation tool and Framework For Misbehavior Detection (F2MD) [25]. Even though the dataset is simulated, the implemented scenario is based on vehicle traces provided by the Luxembourg SUMO Traffic (LuST) scenario and validated with real traffic data. As a first step, VeReMi

TABLE I: Teacher model training parameters.

Parameters	Teacher's Value
Training set size	90%
Learning rate	0.001
Optimizer	Stochastic gradient descent (SGD)
Loss Function	Cross Entropy
Batch Size	32
Training Epochs	550

dataset was regenerated with F2MD. The generated dataset contains 24 folders for each hour and each folder contains a log file for each vehicle. Along with logs of each vehicle, there is also a ground truth file that contains the messages sent from every vehicle to the network. Only the ground truth file is used as it contains already all of the messages from every vehicle. Normally, the data in the ground truth file is not labeled. Therefore, it must re-match with each individual attack label from each vehicle's log files. To avoid re-matching every vehicle log file with the ground truth files, we added the sender attack type directly to F2MD framework during data generation processes. At the end of data generation, the primary information each ground truth file contains was the vehicles' position, heading, speed, acceleration, timestamp and it's identification.

As defined previously, the IDS setting in front of the application will handle a continuous stream of data. As a result, already generated static data needs to be transformed into a continuous stream of data as if they are being received from other vehicles. To do so, the initial step is to separate and group each log by the sender, then sort them by timestamp. Because the data separated by each vehicle is time-series data, it can be segmented into windows. For windows creation, we opted for a window of size 20 with a step size of 10. At this step, every single message in the dataset was labeled as an attack. This is not ideal for classifying a window of 20 values, hence the entire window was labeled rather than a single message. There are 19 types of attacks in the VeReMi extension dataset. Thus, there are 19 classes for the attacks and one for the normal class, for a total of 20 classes. Each window is then assigned a probability for each attack and the normal class. Another important part of model construction is feature selection. For this dataset, nine features were chosen including message receive time and (x, y) coordinates for the position, heading, acceleration, and speed. The input of the model is a matrix of size $20 * 9$, where 9 is the number of selected features and the output is a vector of size $20 * 1$, the probability for each class. Finally, there were a total of 1.75 million labeled windows. Now, if a window is presented to our model, it will be the same situation for every 10 data points received from other vehicles classifying the previous 20 values. For the first classification, one must wait for the first 20 data points received. Before giving the window to the model, the data were normalized with a standard scalar using the following operation $z = (x - u)/s$ where z is the normalized data, x is the input data, u is mean and s is the standard deviation.

C. Teacher model implementation, training, and evaluation

Once the dataset pre-processing is complete, the training of the teacher model can start. For this part, we used results published by the authors of [13]. They proposed two types of classification: sequence classification and sequence image classification. They found out that for the VeReMi extension dataset, sequence image classification with CNNs gives the best results in terms of accuracy and prediction time compared to other architectures using LSTM, CNN-LSTM, and MLP. Considering their findings, sequence image classification with the CNN model was implemented in this work. In their method and before passing the data to any DL model, they convert them to images. Then, images are sent to DL models for classification. For our dataset, the image generation step is already done as we can simply consider our window of 20x9 as an image of 20x9 pixels and it has already been normalized in the previous step.
For the teacher model, we opted for the DL architecture illustrated in Figure 2. The model is constructed by four convolutional layers followed by two linear layers.

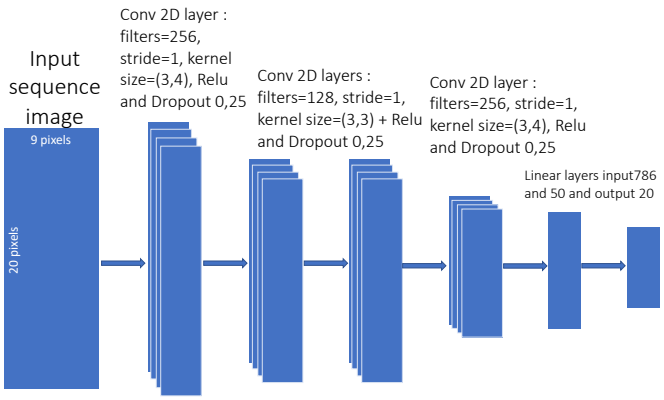

Fig. 2: Teacher CNN architecture

After defining the architecture, the model was trained with the parameters of Table I. The teacher model was let to train longer than the student models to maximize accuracy and extract the most amount of knowledge possible about the different kinds of attacks. Consequently, the teacher model was trained for 550 epochs and reached the performance in Table II.

TABLE II: Teacher and best student model evaluation results.

Metrics	Teacher	Student (sm1)	Optimization
Accuracy	98%	97%	-1%
Recall	98%	97%	-1%
Precision	98%	96%	-2%
f1 score	98%	97%	-1%
Prediction time	1.12ms	0.52ms	+53%
Total parameters	534k	92K	+82%

D. Student model implementation, training, and evaluation

Once the teacher training is done, the training of the student model can start. As illustrated in Figure 3 and Figure 4, two

different architectures sm1 and sm2 were tested. Both of these architectures are with two layers of CNN followed by two linear layers. To train the models, the loss is calculated with the function defined in Equation 2. In addition, to investigate the influence of parameters on the model performance, namely the size of the training dataset and the parameters t, and α used in the loss function. The training dataset sizes are calculated as a proportion of the teacher's training dataset. Table III gives the values for training dataset sizes, t, and α.

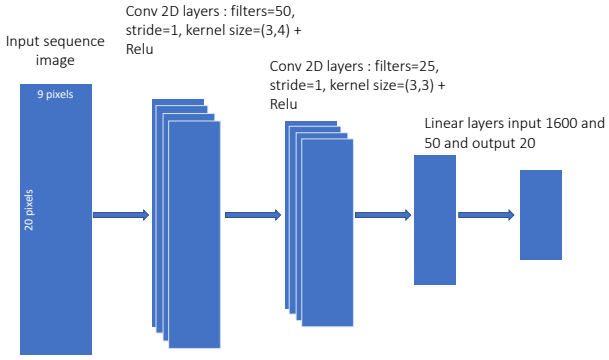

Fig. 3: Student CNN architecture (sm1).

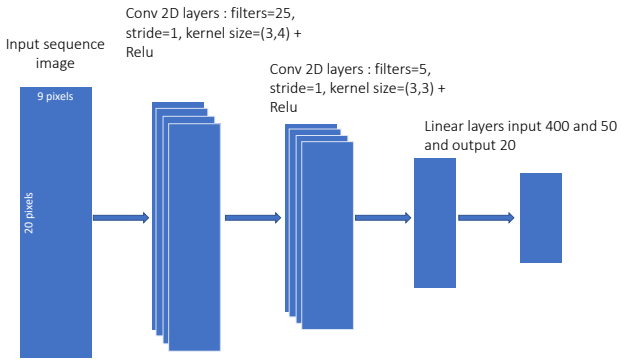

Fig. 4: Student CNN architecture (sm2).

TABLE III: Student models training parameters.

Training dataset size	20%, 15%, 10%, 5%, 3%, 2%, 1%
t **for the loss function**	20, 10, 5
α **for the loss function**	0.9, 0.8, 0.7, 0.6, 0.5
batch size	32
Learning rate	0.001

To evaluate the student model, only the accuracy score was plotted in Figure 5. For the evaluation, the test sets of the student and the teacher remained unchanged. We obviously notice bigger dataset with more complex models has better performance in terms of accuracy. However, other parameters such as t and α play also a vital role in tweaking the performance. Figure 5 shows that for the same model larger values of t give better results on average. As has been already discussed earlier, bigger values of t produce softer outputs for the models and are easier for students to match them. For the

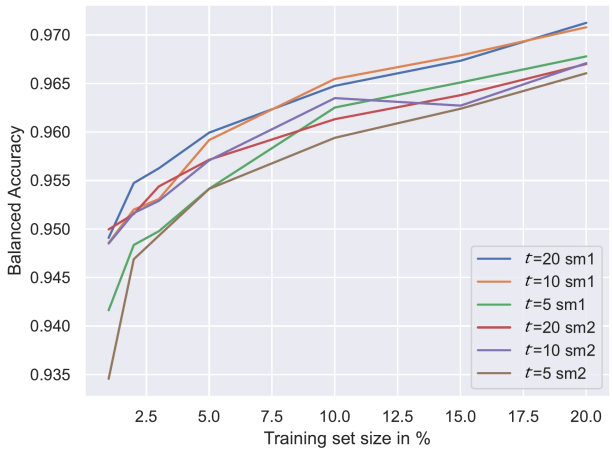

Fig. 5: Student models accuracy for different values of t with $\alpha = 0.9$

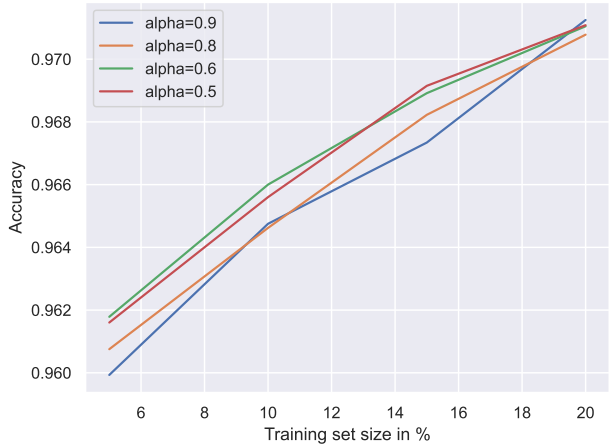

Fig. 6: Student models accuracy with for different values of α with $t = 20$ and model architecture sm1

parameters, α dataset sizes from 5% to 20% were tested as datasets less than 5% degrades considerably the performance.

Figure 6 illustrates that for The VeReMi Extension dataset, the parameter α has a substantial impact and causes around 1% decline in accuracy. When higher weight is given to the teacher's output as the dataset size increases, the student models provide slightly better outcomes. While near-equal weights on the teacher output and true output produce superior outcomes for smaller datasets.

E. Main findings and discussions

Overall, with the best parameters for the sm1 model, the decrease in accuracy is only 1% with 20% of training data. Yet it has only 92K parameters, which is 17% percent of teachers' parameters. Consequently, for the prediction time, the student model takes 0.52ms which is only 46% of the time

required by the teacher model. The above-mentioned findings clearly indicate the trade-off between a model's complexity and performance while also demonstrating the impact of α, t, and training dataset size.

V. Conclusion

Securing the 5G-V2X network efficiently is a critical issue, and numerous attempts are being made to meet the expectations, on top of the emerging network slicing paradigm. In this paper, we investigated a new scheme to effectively deploy an IDS harnessing Knowledge Distillation for deep learning models to detect attacks in 5G-V2X sliced networks. Through in-depth experiments on a recent dataset, we demonstrated the feasibility of the KD concept to provide lightweight DL-based IDS without hurting attack detection accuracy. In future work, we plan to further improve the models to attain the highest level of accuracy and reduce the model size by using other advanced techniques in deep Learning, DL model compression, and acceleration domain.

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] "System architecture for the 5g system," *3GPP TS 23.501*, vol. 15.1.0 Stage 2, 03/2018.

[2] A. A. Kadhim, "5g and next generation networks," in *2018 Al-Mansour International Conference on New Trends in Computing, Communication, and Information Technology (NTCCIT)*. IEEE, nov 2018. [Online]. Available: https://doi.org/10.1109%2Fntccit.2018.8681173

[3] J. N. R. Lu, L. Zhang and Y. Fang, "5g vehicle-to-everything services: Gearing up for security and privacy," *in Proceedings of the IEEE*, vol. 108, no. 2, pp. 373–389, 2020.

[4] T. E. Abdelwahab Boualouache, "A survey on machine learning-based misbehavior detection systems for 5g and beyond vehicular networks," 2021.

[5] R. W. van der Heijden, S. Dietzel, T. Leinmüller, and F. Kargl, "Survey on misbehavior detection in cooperative intelligent transportation systems," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 779–811, 2018.

[6] 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.

[7] S. B. Saad, A. Ksentini, and B. Brik, "A trust architecture for the sla management in 5g networks," in *ICC 2021 - IEEE International Conference on Communications*, 2021, pp. 1–6.

[8] V. N. Sathi and C. S. R. Murthy, "Distributed slice mobility attack: A novel targeted attack against network slices of 5g networks," *IEEE Networking Letters*, vol. 3, no. 1, pp. 5–9, 2021.

[9] S. Ben Saad, A. Ksentini, and B. Brik, "An end-to-end trusted architecture for network slicing in 5g and beyond networks," *SECURITY AND PRIVACY*, vol. 5, no. 1, p. e186, 2022.

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

[11] J. D. Geoffrey Hinton, Oriol Vinyals, "Distilling the knowledge in a neural network," 2015.

[12] T. Alladi, V. Kohli, V. Chamola, and F. R. Yu, "Securing the Internet of Vehicles: A Deep Learning-Based Classification Framework," *IEEE Networking Letters*, vol. 3, no. 2, pp. 94–97, 2021.

[13] T. Alladi, V. Kohli, V. Chamola, F. R. Yu, and M. Guizani, "Artificial intelligence (ai)-empowered intrusion detection architecture for the internet of vehicles," *IEEE Wireless Communications*, vol. 28, no. 3, pp. 144–149, 2021.

[14] H. Grover, T. Alladi, V. Chamola, D. Singh, and K.-K. R. Choo, "Edge computing and deep learning enabled secure multitier network for internet of vehicles," *IEEE Internet of Things Journal*, vol. 8, no. 19, pp. 14 787–14 796, 2021.

[15] T. Alladi, A. Agrawal, B. Gera, V. Chamola, B. Sikdar, and M. Guizani, "Deep Neural Networks for Securing IoT Enabled Vehicular Ad-Hoc Networks," in *ICC 2021-IEEE International Conference on Communications*. IEEE, 2021, pp. 1–6.

[16] T. Alladi, B. Gera, A. Agrawal, V. Chamola, and F. R. Yu, "DeepADV: A Deep Neural Network Framework for Anomaly Detection in VANETs," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 11, pp. 12 013–12 023, 2021.

[17] A. Boualouache, T. E. T. Djaidja, S.-M. Senouci, Y. Ghamri-Doudane, B. Brik, and T. Engel, "Deep learning-based intra-slice attack detection for 5g-v2x sliced networks," in *2022 IEEE 95th Vehicular Technology Conference: (VTC2022-Spring)*, 2022, pp. 1–5.

[18] H. Sedjelmaci, S.-M. Senouci, N. Ansari, and A. Boualouache, "A trusted hybrid learning approach to secure edge computing," *IEEE Consumer Electronics Magazine*, vol. 11, no. 3, pp. 30–37, 2022.

[19] A. Thantharate, R. Paropkari, V. Walunj, and C. Beard, "Deepslice: A deep learning approach towards an efficient and reliable network slicing in 5G networks," in *2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*. IEEE, 2019, pp. 0762–0767.

[20] N. A. E. Kuadey, G. T. Maale, T. Kwantwi, G. Sun, and G. Liu, "DeepSecure: Detection of Distributed Denial of Service Attacks on 5G Network Slicing-Deep Learning Approach," *IEEE Wireless Communications Letters*, 2021.

[21] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, "A survey of model compression and acceleration for deep neural networks," *arXiv preprint arXiv:1710.09282*, 2017.

[22] N. K. S. Abbasi, M. Hajabdollahi and S. Samavi, "Modeling teacher-student techniques in deep neural networks for knowledge distillation," *2020 International Conference on Machine Vision and Image Processing (MVIP)*, p. 2, 2020.

[23] "5g-insight," http://5g-insight.eu/, accessed: 2022-10-31.

[24] J. Kamel, M. Wolf, R. W. van der Hei, A. Kaiser, P. Urien, and F. Kargl, "Veremi extension: A dataset for comparable evaluation of misbehavior detection in vanets," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020, pp. 1–6.

[25] J. Kamel, M. R. Ansari, J. Petit, A. Kaiser, I. Ben Jemaa, and P. Urien, "Simulation framework for misbehavior detection in vehicular networks," *IEEE Transactions on Vehicular Technology*, 2020.