Attack Endgame: Proactive Security Approach for Predicting Attack Consequences in VANET

Mohammed A. Abdelmaguid*, Hossam S. Hassanein*, Mohammad Zulkernine*
* School of Computing, Queen's University, Kingston, Ontario, Canada
Emails:{18ma5, hossam, mzulker}@queensu.ca

***Abstract*—In the fast dynamic environment of Vehicle Ad Hoc Networks (VANETs), proactive security measures are necessary. Reactive security has been VAVNETs' guardian angel for some time, but now it is insufficient against current security attacks. Attack prediction is a promising solution capable of keeping up with the recent cyber security challenges. First, we need to understand where prediction fits in the attack process. To accomplish this, we introduce an attack life cycle in a VANET and exploit the proactive and retroactive phases. One of the proactive phases is the after-effect of the attack or what we call attack endgame. We use the Framework for Misbehavior Detection (F2MD) to simulate an attack effect with adverse side effects on road traffic. We implement traffic warning messages in F2MD. Then, we create attacks on these messages, namely "fake accident", and simulate the effect of these attacks on the vehicles while capturing the results using F2MD. We simulate the impact of acting on these messages or the attack endgame, which manifested in creating hazards. We use Recurrent Neural Network (RNN) models to predict the endgame of the fake accident attack on the road. We experiment with vanilla artificial neural network solutions to create a baseline. Afterward, we use Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU) to build a stacked RNN model to predict the attack endgame at different time windows. They effectively predict the occurrence of a hazard up to 3.5 minutes ahead with over 80% accuracy.**

***Index Terms*—Proactive security, Vehicular Ad hoc Network (VANET), Framework for Misbehavior Detection (F2MD), Machine learning, Neural network, LSTM, GRU.**

I. Introduction

Cyber attacks are evolving in terms of intensity and complexity. Moreover, we are moving into an era where more complex attack strategies are adopted, which makes attack detection an arduous task [1]. Most security approaches deal with the aftermath of the attacks [2]. That being the case, this limits the number of actions that can be taken to overcome the attack, let alone prevent it. As a consequence, attack prediction has become a must [3].

Prediction is an estimation of the outcome of unforeseen data. Attack prediction can aid the network security state with different comprehensive approaches [4]. First, it can alleviate the risk to network security by predicting an attack based on the initial signs or the attack's ultimate goal. Second, it can increase the network's ability to sense and identify tampered messages before the intended damage occurs. Finally, it can help preemptively infer the attack or the attack's next steps. Above all, dynamic and high mobility networks such as Vehicular Ad Hoc Networks (VANETs) suit a proactive approach to securing exchanged messages [5].

In general, vehicles exchange different types of messages in VANET around the clock. Basic safety messages (BSMs) are sent periodically with vehicle-related information such as send time, speed, and position. Vehicles also exchange event-driven safety messages, which are sent when specific safety conditions are met, such as heavy traffic, road construction, or accidents. The work in this paper is based on the security threats to event-driven safety messages [6].

In order to understand where security prediction can be applied during VANET communications, we introduce an attack life cycle for VANET. Researchers from Lockheed Martin were the first to articulate the cyber attack life cycle as a Cyber Kill Chain (CKC) to better understand the overall adversary behavior in cyber attacks. CKC was also created to assist defenders in the decision-making process for averting attacks. We adjust the CKC model to one that can help us understand the attack life cycle in VANET. A specific phase has been introduced in the VANET attack life cycle called "Achieving attack goals." This phase describes the effect of the attack, not the attack itself. We focus on predicting the impact of the attack, which we call the attack endgame.

Attack endgame prediction focuses on the impact of the attack. There is a difference between attack endgame prediction and attack intention recognition. Endgame prediction concentrates on the outcome of the attack, with limited attention to the attack itself. In contrast, intention recognition typically identifies the attack first and then attempts to determine its ultimate purpose. Therefore, intention recognition is more related to multi-step attacks.

In this paper, we propose a novel approach that predicts further impacts of the attacks on VANET. This approach helps the security authorities in VANET to build proactive strategies to alleviate the impact of attacks. To do so, we re-identify the life cycle of attacks on VANET by including the consequences of those attacks. We revisit the underlying implementation of F2MD [7] to accommodate the new specifications of the attacks life cycle in VANET. In particular, we enable F2MD to simulate a vehicle's behavior after receiving malicious messages. The generated scenario represented a misbehaving attack, namely fake accident attack. We also evaluated attack endgame prediction using time series RNN models, and the obtained results showed the possibility of predicting the effect of the attack several minutes before it happens.

ICC 2023 - IEEE International Conference on Communications | 978-1-5386-7462-8/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICC45041.2023.10279188

The rest of the paper is organized as follows. Section II, introduces the attack prediction challenges and discusses the work related to attack prediction in VANET. It also describes the attack life cycle in VANET. Section III introduces the attack endgame and provides a detailed description of the simulation setup, the F2MD framework, and the process of simulating the attack endgame. Section IV highlights the simulation results of the proposed model. Finally, Section V draws the concluding remarks of the paper.

II. Proactive Security in VANET

In this section, we identify the challenges that face attack prediction in VANET, along with related work. Next, we identify where and how proactive security can exist in VANET. We argue that determining where security prediction can be applied depends on our understanding of the attack life cycle. Accordingly, we present our VANET attack life cycle to identify where and how proactive security can take place.

A. Attack Prediction

Proactive security approaches are not widely used due to several challenges [8]. In this section, we will examine three challenges, why these challenges exist, and how to overcome them.

1) Prediction vs. detection: The first challenge is palpable when an introduced solution declares attack prediction while, in fact, the attack-identifying process happens after the attack has taken place. This should not be considered attack prediction since the damage is already done.

2) Dataset for prediction: The second challenge is the absence of a standard dataset for attack prediction. The available datasets are not designed for evaluating security prediction, especially in VANET.

3) Prediction metrics: The third challenge is the lack of clear metrics for evaluating attack prediction. Aside from assessing the accuracy of the prediction, we also require metrics concerning the timing criteria. By timing criteria, we mean the time until the successfully predicted attack will take place [9]. Hence, in our experiments, we introduce time as an essential matrix in the evaluation process since it is a deciding factor between prediction and detection.

Despite the discussed challenges regarding attack prediction in VANET, many researchers still managed to introduce promising proactive solutions [10]. Although different techniques have been proposed, our interest is in RNN-based solutions as these are more related to our work. Dasgupta et al. [10] build a prediction-based spoofing attack detection strategy. They used the LSTM model on a real-world driving dataset, comma2k19 [11]. The dataset consists of various autonomous vehicle (AV) sensor data (e.g., Control Area Network (CAN), Global Navigation Satellite System (GNSS), and Inertial Measurement Unit (IMU) data). Examples of these collected sensor data are time, latitude, longitude, car speed, steering angle, and acceleration. They used these data to predict each time stamp's traveled distance between the current and the immediate future locations. They considered one type of spoofing attack in their study where the spoofer sends manipulated GNSS signals to show that the AV is taking an exit while still steering forward. The tested LSTM model exhibited a high detection rate, with an average absolute error of 0.000046 meters and a computational latency of 5 milliseconds, which is in line with the VANET latency constraint of 100 milliseconds. Fang et al. [12] used the LSTM network to create a cyberattack rate prediction framework called BRNN-LSTM. Most often, statistical approaches are used for predicting attacks rate. Still, BRNN-LSTM showed that deep learning or neural networks can outperform the legacy statistical modules like Autoregressive Integrated Moving Average (ARIMA) and Generalized AutoRegressive Conditional Heteroskedasticity (GARCH). BRNN-LSTM was tested on five real-world datasets, and except for only one, the model can be used without retraining. LSTM was also used to predict spoofing attacks [10]. Tested on a public real-world data set called comma2k19, LSTM was used to predict the distance between two consecutive vehicles. The spoofing attack was detected in real-time using an error threshold, which was determined by calculating the vehicle's immediate and future location and the GNSS device's positioning error.

B. VANET Attack Life Cycle

We introduce our proposed version of the attack life cycle in VANET to present where proactive security can be applied through the cycle. Figure 1 displays a comparison between the cyber attack life cycle phases against the VANET attack life cycle. The figure also specifies what phases belong to proactive security and what should belong to reactive security.

As shown in Figure 1, reconnaissance is the first phase in the cyber attack life cycle, and it is the same in VANET. In the reconnaissance step, the attacker develops a target for their attack. The second step, weaponizing, is when the attacker creates the payload of the attack. Whereas in VANET, this is where the attacker begins to create a false message (either BSM or traffic warning message). The third step, deliver, is where the vulnerability in the system is weaponized. In VANET, this is where the attacker exchanges false messages by broadcasting them to the surrounding vehicles. The fourth step, exploit, is where the weaponized payload is executed. However, in VANET, this is where the vehicles start to act upon the received message. The remaining steps, control, execute, and maintain, are more related to a typical cyber attack than VANET. In these steps, the attackers try to escalate their privileges and maintain their presence in the system by creating back doors. Whereas in VANET, we have what are called attack goals or effects.

The attack goals are the attack's impact on the vehicles that acted on the received false messages. Attack goals can create a hazard, an accident, or take over control of the vehicle [13]. To achieve the attack goals, the vehicle has to receive the false message and behave accordingly. In VANET, receiving the attacker payload (i.e., the false message) does not imply that the payload fulfilled its purpose. Only when the vehicles

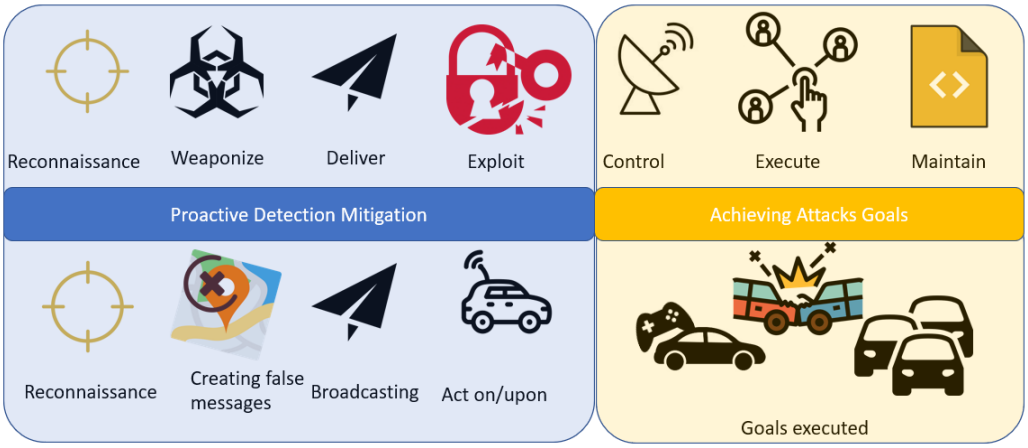

Figure 1: Introducing attack life cycle in VANET

start to act upon the incorrect information, we can say that the exploit phase is succeeded.

III. Simulating Attack Endgame

By mapping the attack life cycle in VANET over the cyberattack life cycle, we segregate the boundary between proactive and reactive attack steps in the VANET attacks. The "Act on/upon" step in Figure 1 separates where proactive solutions operate from reactive ones. Hence, we consider any applied defending security techniques beyond this step will be more for detection purposes than prediction purposes. Owing to the fact that once the attack achieves its goal, any provided solutions should be considered reactive.

We propose the idea of predicting the attack endgame, which entails creating a hazard. Below, we discuss the detailed steps for creating the simulation environment for generating an attack endgame.

A. Simulation Setup

We implement our simulation using Instant Veins version 5.2, a preconfigured Linux instance that comes with Veins installed [14]. A VANET simulation generally consists of two types of simulators: a traffic simulator and a network simulator. We use Simulation of Urban Mobility (SUMO) [15] as the traffic simulator. We use vehicles in network simulation (Veins) and OMNet++ [14] for the network simulator. OMNeT++ is responsible for laying down the infrastructure and providing the needed tools for writing a simulation. All the above modules and frameworks are included in the F2MD, which we will discuss in detail.

B. Framework For Misbehavior Detection (F2MD)

Framework for Misbehavior Detection (F2MD) [7] is a framework that provides different functionalities and capabilities for researchers in the domain of VANET security. One of these functionalities is simulating attacks in VANET. F2MD was used before to generate the VeReMi dataset [16] and its extension [17]. The F2MD developers made the source code publicly available, so other researchers could add to it and explore new research areas [18]. We use F2MD to add the effect of the attack.

In this subsection, we present the changes that we made on F2MD in general and the changes that were done to some modules in specific to generate the attack endgame.

1) Used scenario: The Luxembourg traffic scenario (LuST) [19] was used to provide a realistic and holistic scenario. We conduct the experiment on the actual LuST scenario presented in F2MD to provide a simulation as generic as possible since it allows for a perfect balance in terms of size and representation. The LuST was built according to an actual mid-size European city (Luxembourg). The included traffic demand is also based on realistic information.

2) Attack module: We generate a new attack called the fake accident attack. We have two types of cars in our simulation: benign and malicious. The malicious car sends a traffic warning message about a fake accident with the road number. Furthermore, to make the attack as authentic as possible, we ensured that the attacker's car did not always send an attack message. Accordingly, we add a random number generator where the attack car sends an attack message only if the generated number is bigger than a certain threshold. The random number is between 0 and 100, and we choose 70 as the threshold based on the different experiments to ensure that the attack is represented just well enough. Another reason for choosing 70 as a threshold is that the percentage of traffic warning messages added to the F2MD is lower than all exchanged messages. In our simulation, the cars exchange Basic Safety Messages (BSMs) more frequently than traffic warning messages. Given that not all traffic warning messages we implement are attack messages, we need to increase the chance of the random occurrence of the attack. Therefore, to achieve this attack's goal in the first place, we have to keep searching for the suitable probability of the attack occurring and choose this threshold. Otherwise, we will end up with very few attack messages to generate a hazard or too much hazard, making hazard a normal behavior in our network.

3) General simulation parameters: We kept the simulation parameters the same except for the attack probability. This

one is different from the attack threshold introduced in the attack module. The attack probability parameter is related to the overall attack probability in the network. In contrast, the previously introduced attack threshold was related to the malicious vehicles themselves (i.e., whether the vehicle sends an attack message or not). That being the case, we had to increase the attack probability from 0.05 to 0.3 for the same reasons discussed previously in the attack module.

C. Fake Accident Attack

Originally, F2MD was built with the aim of encouraging other researchers to contribute to the framework with new types of VANET attacks [16]. The previously introduced set of attacks were all basic safety message (BSM) type-related attacks. In our work, we introduce a new type of attack related to event-driven safety messages called the fake accident attack.

There are different types of exchanged messages in VANET. BSMs are sent periodically, and event-driven messages are sent in case an event triggers them. Attack endgame prediction simulation files contain records of both BSM and traffic warning messages. We simulate one type of event: reporting an accident on a specific road number. In particular, vehicles send event-driven messages to report an accident. In our simulations, we capture the effect of receiving event-driven messages on vehicle behavior. We simulate the impact of receiving an accident message, whether benign or malicious, according to the description in the "F2MDVeinsApp" module.

In a fake accident attack, some malicious vehicles send an event-driven message to report a fake accident. This would lead to consequences such as vehicles slowing down and may even cause a hazard if the number of affected vehicles exceeded a specific limit. A hazard, in our case, takes effect when three conditions are satisfied: (1) The vehicle is on the reported road; (2) The speed of the vehicle goes below 20% of the maximum speed of the road; (3) The number of affected vehicles become more than a certain threshold (five vehicles in our simulations).

Figure 2 shows a zoomed-in figure for the occurrence of a hazard over a period of time. The figure shows spikes in the frequency of hazards at specific points in time, with each spike representing a hazard occurrence. Additionally, the figure shows how we can create a hazard and then allow the simulation to get back to its normal state multiple times without crashing. Our simulation also provides an indicator of the hazard occurrence time. It's important to note that the time at which the attack message was sent will be ahead of the actual occurrence of the hazard. This is because it takes time for the hazard to materialize once the malicious message has been received. Figure 3 provides an overview of the events that transpired during the simulation period. The binary notation used in the three graphs assigns a value of 1, representing the occurrence of a hazard at a specific time, a value of 0, indicating that no hazard was detected at that time, and a value of -1, indicating the presence of an unidentified event. A thorough analysis and discussion of the contents of Figure 3 will be presented in section IV-B

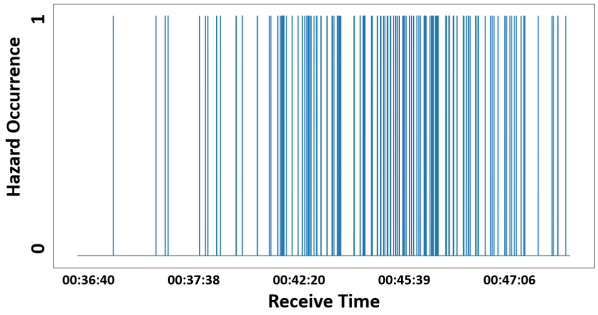

Figure 2: Occurrence of hazards over time: a zoomed-in view

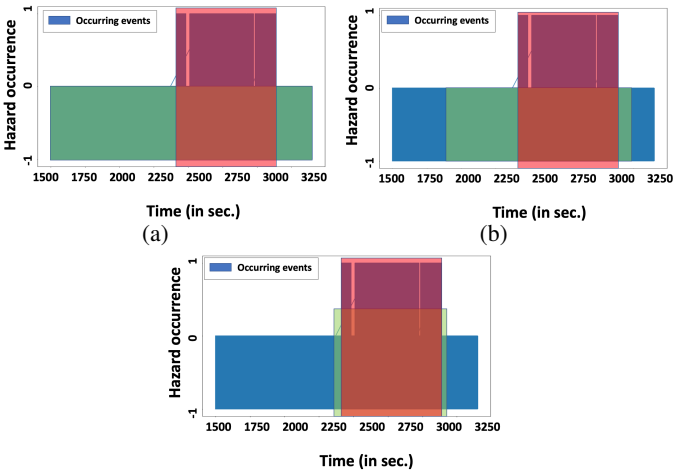

Figure 3: Assessment of the impact of data scope on hazard prediction, with each subfigure displaying a different scenario in which a different amount of data is included

IV. Experiments and Results

In our experiments, we use recurrent neural network (RNN). RNN is incorporated with memory for storing information from prior inputs and using it to influence both input and output. In our case, we believe that input and output are not independent of each other, which means memory-based models such as RNN are more suitable in our situation.

In the following experiments, we try to predict the effect of a false traffic warning message on the network. At a designated time, malicious cars will transmit a falsified traffic alert. However, the full ramifications of this false information may take some time to become apparent. The effect we look at here is the resulting hazard from acting upon the wrong information received from the malicious car.

We conduct three experiments to evaluate attack endgame prediction modules. We consider the results obtained by the LSTM and GRU models as a baseline. The first two experiments create a baseline for NN models. The first experiment is for testing how far we can predict the future. The second experiment is conducted to test the correlation between the type of carried-on information from past interactions and prediction results. Finally, the third experiment is conducted to evaluate our proposed model against the baseline.

We implement a cross-validation approach in all our models using the Train-Valid-Test split technique. Moreover, since we are dealing with a time series problem, we have to use the temporal component splitting technique since a random split could mess up the chronological order of the events. We use 80% of the data for training, 10% for validation, and 10% for testing. Our regression models have either LSTM or GRU layer followed by one "relu" dense layer.

To evaluate the models' effectiveness, we use an R-squared score. Higher R-squared values indicate that the difference between the observed data and the fitted values is small. In other words, a high R-squared suggests that the model's variance is similar to actual values and how strongly related they are.

A. Experimenting with How Far Can We Predict?

The first experiment is about the relation between prediction time and obtained accuracy. The obtained records from the simulation are indexed by the received message time. The difference between each message is one second, which reflects the difference between every record. Every message acts as an indicator of the network state at the time when the message is sent. The network has three states: hazard, no hazard, or unknown. The goal of the experiment is to predict the occurrence of a hazard.

In our experiments, we have to decide on two numbers, n, which is the window size, and r, which is the step size. In order to test the model's ability to predict an event after n seconds, we take a window of n messages, and then we predict the message $n + 1$. After that, the window moves with the given step. For window sizes between 30 - 90, we use one step (i.e., shifting by one message), and for window sizes bigger the 90, we use ten steps. We have to use bigger steps with bigger window sizes since the experiments crashed with a such small steps in big windows. We also find this compatible with the nature of the problem. The farther the prediction goes, the less distant data from the actual event we will need. In other words, the older the data, the less relevant it becomes to predictions.

We use vanilla LSTM and GRU to create a baseline and to see how far we can predict in the future and with what accuracy. As shown in Figures 4, 5, and 6, the obtained results show that we could maintain performance over 80% in predicting the occurrence of a hazard up to 3.5 minutes ahead. We consider these auspicious results since 3.5 minutes can give enough time for counterargument solutions to act. The model's performance could be improved, but the algorithms used are primitives, which offers much room for improvement, as demonstrated by the third experiment.

B. Testing the Effect of Data Quality on the Prediction Process

The objective of the second experiment is to examine the impact of varying time frames on the predictive accuracy of the model. To achieve this, we divide the overall time frame into three segments, as depicted in Figure 3. The subfigures showcase each time segment, with each subfigure including a different amount of data. The difference between

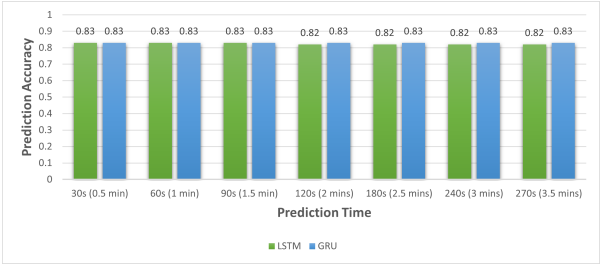

Figure 4: Prediction results using LSTM and GRU from 30 secs up to 3.5 mins on all available data

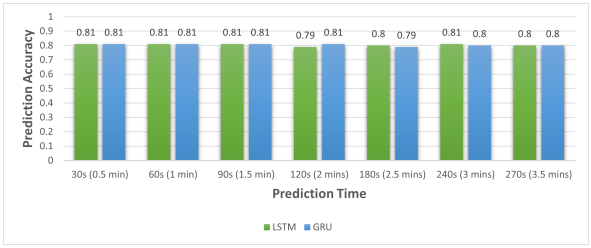

Figure 5: Prediction results using LSTM and GRU from 30 secs up to 3.5 mins on smaller time window

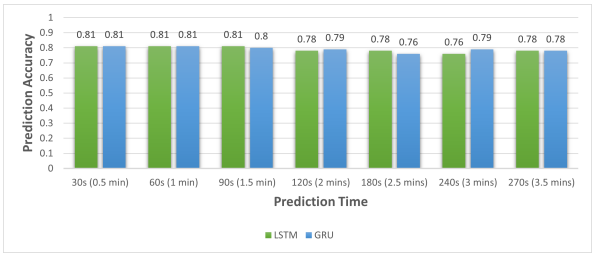

Figure 6: Prediction results using LSTM and GRU from 30 secs up to 3.5 mins on only hazard data

the subfigures lies in the inclusion of messages that are further from the hazard occurrence time. Figure 3(a) represents the scenario in which all available data is included, Figure 3(b) includes less data while still capturing information beyond the hazard occurrence duration, and Figure 3(c) encompasses data only around the hazard occurrence time. The green highlighted areas in every subfigure represent the three different included time segments in each case, while the red highlighted areas signify hazards. Testing the model's accuracy on each time segment improves our understanding of the time-performance relationship.

In our experiments, we thoroughly evaluate each scenario depicted in Figure 3 and obtain insightful results. The findings are showcased in Figures 4, 5, and 6, providing a comprehensive overview of our results. These figures show the predictive accuracy of the model across the different time segments and provide insights into the impact of the time segment on the model's performance. By looking at Figures 4, 5, and 6, we can see a decline of the overall accuracy of the results where a number of them fall under 80%. Hence, it shows how the quality of the used data has a noticeable effect on the prediction results. The difference between the first, second, and third segments has two factors: the duration and

the incorporated information. The first factor shows that not having enough time before the occurrence of the event (i.e., the hazard) can affect the prediction process severely. The second factor demonstrates the importance of including past recorded events responsible for creating the hazard. Consequently, the second factor affects the attack endgame predicting process. As shown in Figure 3, these events occurred when the actual attack took place, and they are important to include even though the hazard only materializes later.

C. Comparing Stacked LSTM and GRU Model Versus Generated Baseline

Time before hazard	LSTM	GRU	Stacked LSTM & GRU
30s (0.5 min)	0.81%	0.81%	0.84%
60s (1 min)	0.81%	0.81%	0.84%
90s (1.5 min)	0.81%	0.8%	0.84%
120s (2 min)	0.78%	0.79%	0.83%
180s (2.5 min)	0.78%	0.79%	0.82%
240s (3 min)	0.76%	0.79%	0.81%
270s (3.5 min)	0.78%	0.78%	0.81%

Table I: Comparison between the obtained results from the LSTM, GRU, and the stacked model

The purpose of this experiment is to introduce a new stack model. In this model, we stack an LSTM and a GRU model, and we compare the obtained results against both previously introduced Vanilla LSTM and Vanilla GRU. LSTM is preferred in the case of large data, while GRU is recommended in the case time constrained problems [20]. Since VANET undergoes time constraints, and at the same time, we can accumulate a large number of data over time, we decided to stack both LSTM and GRU.

We conduct this experiment only on the last data frame, where only the data around the hazard is considered. As discussed in the second experiment, this data frame is the one with the least accuracy values. In this experiment, we try to demonstrate the benefit of using the introduced stacking model in improving the obtained results. Table I shows a comparison between the obtained results by all three models. We can see an increase in the R-square values in all different time windows. The figure shows how all the obtained results managed to break the 80% limit with the new model.

V. Conclusion

Proactive security approaches are a natural evolution of the current widely used reactive methods. This paper proposes one of the first steps toward adopting proactive security techniques in Vehicle Ad Hoc Networks (VANETs). Our work shows how we could predict the effect of a cyber attack. We used a framework for misbehavior detection (F2MD) to generate attacks in VANET, and we incorporated a traffic warning message into the F2MD. Next, a fake accident attack was created and deployed, simulating the effect of the attack on the network. We used the results from two RNN models as a baseline and introduced a stacked model composed of LSTM and GRU models. The final results show that the attack endgame could be predicted up to 3.5 minutes prior with 80% accuracy. While the presented work provides insight into the potential effects of an attack, it is not sufficient on its own to fully address the issue as it is essential to also focus on developing countermeasure solutions to mitigate the predicted effects of the attack. Thereby saving lives on the road by avoiding the catastrophic effects of VANET cyber attacks beforehand.

References

[1] Y. Li and Q. Liu, "A comprehensive review study of cyber-attacks and cyber security; emerging trends and recent developments," *Energy Reports*, pp. 8176–8186, 2021.

[2] S. S. Bhuyan, U. Y. Kabir *et al.*, "Transforming healthcare cybersecurity from reactive to proactive: current status and future recommendations," *Journal of Medical Systems*, pp. 1–9, 2020.

[3] O. Ben Fredj, A. Mihoub *et al.*, "Cybersecurity attack prediction: a deep learning approach," in *13th International Conference on Security of Information and Networks*, 2020, pp. 1–6.

[4] M. Husák, J. Komárková *et al.*, "Survey of attack projection, prediction, and forecasting in cyber security," *IEEE Communications Surveys & Tutorials*, pp. 640–660, 2018.

[5] B. Amar Bensaber, C. G. Pereira Diaz, and Y. Lahrouni, "Design and modeling an adaptive neuro-fuzzy inference system (anfis) for the prediction of a security index in vanet," *Journal of Computational Science*, p. 101234, 2020.

[6] B. M. Mughal, A. A. Wagan, and H. Hasbullah, "Efficient congestion control in vanet for safety messaging," in *International Symposium on Information Technology*, 2010, pp. 654–659.

[7] J. Kamel, M. R. Ansari *et al.*, "Simulation framework for misbehavior detection in vehicular networks," *IEEE Transactions on Vehicular Technology*, pp. 6631–6643, 2020.

[8] M. Husák, J. Komárková *et al.*, "Survey of attack projection, prediction, and forecasting in cyber security," *IEEE Communications Surveys & Tutorials*, pp. 640–660, 2018.

[9] H. A. Kholidy, A. Erradi *et al.*, "A finite state hidden markov model for predicting multistage attacks in cloud systems," in *2014 IEEE 12th International Conference on Dependable, Autonomic and Secure Computing*, 2014, pp. 14–19.

[10] S. Dasgupta, M. Rahman *et al.*, "Prediction-based GNSS spoofing attack detection for autonomous vehicles," *CoRR*, 2020.

[11] H. Schafer, E. Santana *et al.*, "A commute in data: The comma2k19 dataset," *arXiv:1812.05752*, 2018.

[12] X. Fang, M. Xu *et al.*, "A deep learning framework for predicting cyber attacks rates," *EURASIP Journal on Information Security*, 2019.

[13] I. A. Sumra, I. Ahmad *et al.*, "Behavior of attacker and some new possible attacks in vehicular ad hoc network (VANET)," in *2011 3rd International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, 2011, pp. 1–8.

[14] C. Sommer, R. German, and F. Dressler, "Bidirectionally coupled network and road traffic simulation for improved IVC analysis," *IEEE Transactions on Mobile Computing*, pp. 3–15, 2011.

[15] D. Krajzewicz, J. Erdmann *et al.*, "Recent development and applications of sumo-simulation of urban mobility," *International Journal on Advances in Systems and Measurements*, pp. 128–138, 2012.

[16] R. W. van der Heijden, T. Lukaseder, and F. Kargl, "Veremi: A dataset for comparable evaluation of misbehavior detection in vanets," *International Conference on Security and Privacy in Communication Systems*, pp. 318–337, 2018.

[17] J. Kamel, M. Wolf *et al.*, "VeReMi Extension: A Dataset for Comparable Evaluation of Misbehavior Detection in VANETs," *ICC 2020-2020 IEEE International Conference on Communications (ICC)*, pp. 1–6, 2020.

[18] M. A. Abdelmaguid, H. S. Hassanein, and M. Zulkernine, "Samm: Situation awareness with machine learning for misbehavior detection in vanet," in *Proceedings of the 17th International Conference on Availability, Reliability and Security*, 2022, pp. 1–10.

[19] L. Codeca, R. Frank, and T. Engel, "Luxembourg sumo traffic (lust) scenario: 24 hours of mobility for vehicular networking research," in *2015 IEEE Vehicular Networking Conference (VNC)*, 2015, pp. 1–8.

[20] P. T. Yamak, L. Yujian, and P. K. Gadosey, "A comparison between arima, lstm, and gru for time series forecasting," in *Proceedings of the 2019 2nd International Conference on Algorithms, Computing and Artificial Intelligence*, 2019, pp. 49–55.