A Multi-Resolution Simulation Platform for Transportation System Security Testing and Evaluation

Final Report

by

Yiheng Feng
Purdue University

Satish Ukkusuri, Purdue University
Hadi Amini, Florida International University
Kemal Akkaya, Florida International University

June 2025

NATIONAL CENTER FOR TRANSPORTATION CYBERSECURITY AND RESILIENCY (TraCR)

DISCLAIMER

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated in the interest of information exchange. The report is funded, partially or entirely, by the National Center for Transportation Cybersecurity and Resiliency (TraCR) under Grant No. 69A3552344812 and 69A3552348317 which is headquartered at Clemson University, South Carolina, USA, from the U.S. Department of Transportation's University Transportation Centers Program. The U.S. Government assumes no liability for the contents or use thereof.

CONTACTS

For more information:

Yiheng Feng
550 Stadium Mall Drive
West Lafayette, IN 47906
Phone: 765-496-5025
Email: feng333@purdue.edu

TraCR
Clemson University
One Research Dr
Greenville, SC 29607
tracr@clemson.edu

ACKNOWLEDGMENT

This work is based upon the work supported by the National Center for Transportation Cybersecurity and Resiliency (TraCR), a U.S. Department of Transportation National University Transportation Center headquartered at Clemson University, Clemson, South Carolina, USA. Any opinions, findings, conclusions, and recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of TraCR. The U.S. Government assumes no liability for the contents or use thereof. The authors would like to thank Dr. Jonathan Petit and Mr. Raashid Ansari from Qualcomm for their technical support in the VASP platform. The views presented in this paper are those of the authors alone.

Technical Report Documentation Page

1. Report No. 4	**2. Government Accession No.** N/A	**3. Recipient's Catalog No.** N/A
4. Title and Subtitle A Multi-Resolution Simulation Platform for Transportation System Security Testing and Evaluation		**5. Report Date:** June 2025 **6. Performing Organization Code:** N/A
7. Author(s) Yiheng Feng, Ph.D.; https://orcid.org/0000-0001-5656-3222 Jun Ying, Ph.D.; https://orcid.org/0009-0005-4608-1883 Eunhan Ka, Ph.D.; https://orcid.org/0000-0003-0954-8075 Satish V. Ukkusuri, Ph.D.; https://orcid.org/0000-0001-8754-9925 Hadi Amini, Ph.D.; https://orcid.org/0000-0002-2768-3601 Kemal Akkaya, Ph.D.; https://orcid.org/0000-0002-7103-4545		**8. Performing Organization Report No.** 4
9. Performing Organization Name and Address National Center for Transportation Cybersecurity and Resiliency (TraCR), Clemson University, 414 A One Research Dr, Greenville, SC 29607 Purdue University, 610 Purdue Mall, West Lafayette, IN 47907 Florida International University, 11200 SW 8th Street, Miami, FL 33199		**10. Work Unit No.** N/A **11. Contract or Grant No.** 69A3552344812 and 69A3552348317
12. Sponsoring Agency Name and Address U.S. Department of Transportation, Office of the Assistant Secretary for Research and Technology, 1200 New Jersey Avenue, SE, Washington, DC 20590		**13. Type of Report and Period Covered** Final Report, 01/01/2024 - 05/31/2025 **14. Sponsoring Agency Code** OST-R
15. Supplementary Notes Conducted under the U.S. DOT Office of the Assistant Secretary for Research and Technology's (OST-R) University Transportation Centers (UTC) program.		
16. Abstract Due to the nature of transportation cybersecurity problems, it is usually too risky to evaluate the attack models in real-world environments. Therefore, it is necessary to develop a simulation environment to validate the proposed models and evaluate the impact on the transportation system in terms of safety and mobility. This project develops realistic simulation environments that include 1) supporting testing of various connected and automated transportation applications from individual vehicle level, traffic level, to the network level; and 2) integrating different attack models and evaluating the impact of certain cyber-attacks on connected and automated transportation applications. Toward these goals, we develop various APIs for the simulation environments that incorporate data spoofing attacks, route guidance attacks, and federated learning-based security.		
17. Keywords Cybersecurity; Simulation Platform; Data Spoofing Attack; Route Guidance Attack; Federated Learning	**18. Distribution Statement** No restrictions.	

19. Security Classif. (of this report) Unclassified	**20. Security Classif. (of this page)** Unclassified	**21. No. of Pages** 33	**22. Price** N/A

Form DOT F 1700.7 (8-72) Reproduction of completed page authorized

TABLE OF CONTENTS

List of Tables

List of Figures

EXECUTIVE SUMMARY

The objectives of this project are to develop realistic simulation environments that include 1) supporting testing of various connected and automated transportation applications from individual vehicle level, traffic level, to the network level; and 2) integrating different attack models and evaluating the impact of certain cyber attacks on connected and automated transportation applications. Toward these goals, we develop various APIs under multiple simulation environments. At the vehicle level, our project utilizes CARLA's capabilities for generating realistic sensor data to further analyze the dynamic impact of adversarial attacks. Furthermore, fourteen different federated learning (FL) aggregation algorithms were evaluated under seven categories of attacks toward sensor data. Results show that the performance of different aggregation algorithms varies substantially across attack categories. At the traffic level, we implement two types of attacks, namely estimated time of arrival (ETA) attack and, multi-sensor fusion (MSF) attack into the VSAP simulation platform and develop APIs for users to customize their own attack scenarios. The MSF attack model is validated using 699 trajectories, and the proposed trajectory-classification method achieved 100% accuracy in distinguishing different attack patterns. Finally, at the network level, a bounded rational route guidance attack (RGA) API is developed, which exploits the trust that travelers place in navigation systems by manipulating routing information, thereby redirecting vehicles onto suboptimal or inefficient routes. Simulation results show that GPS spoofing-based RGAs can significantly increase the network congestion level and vehicle delay.

CHAPTER 1

Introduction

1.1 Research Gaps

Due to the nature of transportation cybersecurity problems, it is usually too risky to evaluate the attack models in real-world environments. Therefore, it is necessary to develop a simulation environment to validate the proposed models and evaluate the impact on the transportation system in terms of safety and mobility. However, existing simulation tools suffer from the following limitations, including

- Single modality: For example, Simulation for Urban MObility (SUMO) [1] is a popular microscopic traffic simulation tool but is limited in modeling vehicle-level behaviors (e.g., perception, and path planning) and V2X communication network.
- No cybersecurity component: Some simulators, such as [2], integrate multiple levels of simulation details, but they do not have the capabilities to conduct cybersecurity-related simulation analysis.
- No transportation applications: A few simulation platforms developed for cybersecurity research only mimic lower-level security functions but lack simulating transportation applications. For example, the VASP platform mainly focuses on simulating V2X communication network level (e.g., security credential management, V2X messages) attacks [3], but has limited functions in replicating traffic-level applications.

Overall, a security-focused multi-resolution simulation platform that is integrated with various attack and defense services is needed.

1.2 Project Goals and Objectives

The objectives of this project are to develop realistic simulation environments that include 1) supporting testing of various connected and automated transportation applications from individual vehicle level, traffic level, to the network level; and 2) integrating different attack models and evaluating the impact of certain cyber attacks on connected and automated transportation applications. Toward these goals, we develop various APIs for the simulation environments, including

- **Sensor attack API**: Users can manipulate the simulated sensor in the Carla simulation environment.
- **Data spoofing attack API**: Users can manipulate data received from simulated sensors, GPS, and V2X communication channels. For example, falsified location and speed of ego and other vehicles and misleading pixels in the image data.
- **Route guidance attack API:** Users can manipulate the path information and vehicle routing algorithms (e.g., fake traffic jams using multiple smartphones).

1.3 Report Organization

The rest of the report is organized as follows. Chapter 2 introduces the data spoofing attack API implemented on the VASP simulation platform. Chapter 3 introduces the route guidance attack API. Chapter 4 presents the evaluation of the real-world applicability of federated learning (FL)-trained models and the impact of adversarial attacks on autonomous vehicle (AV) systems in the Carla simulation environment. Finally, Chapter 5 provides the conclusions of the project.

CHAPTER 2

Data Spoofing Attack API

2.1 Introduction

Next generation transportation system is envisioned to be equipped with advanced sensors for traffic monitoring and vehicle control, vehicle-to-everything (V2X) devices for real-time communication and edge computing to provide safe and efficient traffic management strategies. These emerging technologies improve the transportation system safety and mobility; however, it also opens a new door for cyber-attack. One major threat to the next generation transportation system is data spoofing attack. In data spoofing attack, the attacker deliberately modifies the sensor input or the broadcast message (i.e., BSMs) to mislead vehicle [4-7] or signal controller [8-10]. In this project, two typical data spoofing attacks are selected, including the estimated time of arrival (ETA) attack and the multi-sensor fusion (MSF) attack, to be modeled and integrated on VASP. The selected attacks target on deteriorating the performance of the infrastructure (ETA attack) and autonomous vehicle (MSF attack). The impact of the selected attack on transportation system safety and mobility is non-neglectable.

2.1.1 ETA Attack Introduction

The ETA attack has been proven to be a major threat to the connected vehicle-based traffic control systems (CV-TSC), such as the I-SIG system [11]. As demonstrated in Figure 1, the intersection contains both normal CVs (yellow vehicle) and CVs under attack (red vehicle). When approaching an intersection, both vehicles broadcast BSMs, including the vehicle's current speed and position to the infrastructure. The infrastructure collected BSMs broadcast by vehicles to generate an optimized signal timing plan. When the vehicle is not under attack, its longitudinal movements adhere to car following models, such as the Intelligent Driver Model (IDM) and transmit BSMs that accurately reflect its true trajectory. When the vehicle is compromised by an attacker, its physical movement continues to follow the same car-following behavior, as shown in the blue color, but broadcasts falsified BSMs to achieve the attack goal. For instance, falsified BSMs may indicate unnecessary deceleration, which is different from the actual vehicle movement (shown in red). The ETA of the falsified BSM trajectory is then longer than the true ETA. Consequently, the infrastructure, misled by the incorrect ETA, may generate a non-optimal signal timing plan, such as an unnecessary extension for the current green phase. ETA attack is selected as a typical data spoofing attack to be modeled due to its non-neglectable impact on transportation system mobility.

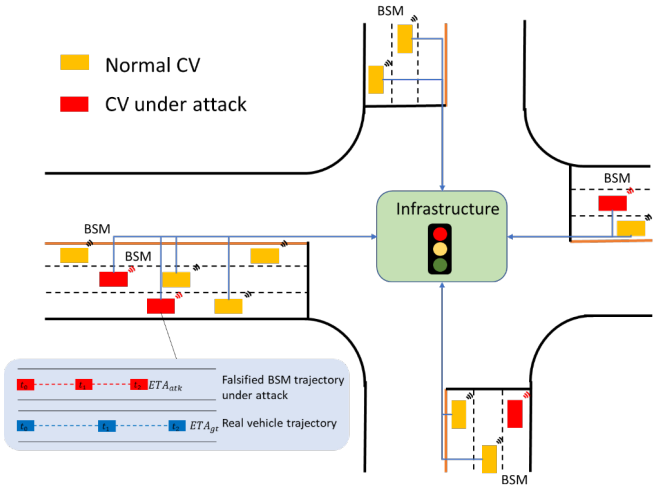

Figure 1: ETA Attack Threat Model

2.1.2 MSF Attack Introduction

Multi-Sensor Fusion (MSF) algorithms are widely used in autonomous vehicles (AVs) to enhance the localization accuracy. It integrates data from multiple sensors, including GPS, IMU (Inertial Measurement Unit) and LiDAR [12,13]. MSF is considered a defense method against spoofing attacks, as compromising all sensors simultaneously is a difficult task. However, a study by Shen et al. [7] demonstrated that manipulating only the GPS receiver's output, the attacker can degrade the MSF algorithm performance significantly. By adding deviation to the original GPS data, the MSF attack can cause large deviations in the MSF algorithm and deviate the vehicle from the road. MSF attack may result in sudden break by following vehicles and even potential collision. Consequently, the MSF attack poses a serious threat to transportation system safety and mobility.

2.1.3 Attack Platform Introduction

The selected attacks are modeled and integrated with the V2X Application Spoofing Platform (VASP) [14], an open source framework developed by Qualcomm to simulate attacks on V2X networks and applications. VASP is developed on VEINS [15], which combines OMNeT++ [16], an event-based network simulator and SUMO, the road traffic simulator. VASP contains 68 typical V2X attacks, covering both kinematic value attacks (i.e., vehicle position, speed, acceleration) and V2X application attacks (i.e., (Intersection Movement Assist (IMA), Electronic Emergency Brake Light (EEBL)). It enables the customization of built-in attacks by adjusting attack parameters such as malicious rate and supports the integration of user-developed attacks into the platform.

2.2 ETA Attack API

2.2.1 Modeling Estimated Time of Arrival (ETA) Attack

2.2.1.1 Model Formula

To model the falsified trajectory under the ETA attack, an optimization model is formulated as shown in **Equation 1**. The generated vehicle trajectory consists of a sequence of trajectory points

at consecutive time steps. Each trajectory point at a given time step includes 5 elements, including $x_t, y_t, v_t, a_t, \psi_t$, which are optimization problem variables. x_t, y_t represent the vehicle's longitudinal and lateral position at time step t. v_t, a_t, ψ_t are vehicle kinematic-related variables, denoting the speed, acceleration, and heading angle at time step t. The total trajectory length is denoted as N, which is determined by the signal update interval.

$$\min_{x_t, y_t, v_t, a_t, \psi_t} \boldsymbol{\theta}^T f(x_t, y_t, v_t, a_t, \psi_t, \boldsymbol{e}_t) \quad (1)$$

s.t.

vehicle safety constraints (**Equations 3-4**)
vehicle dynamic constraints (**Equations 5-13**)

The objective of the optimization function is represented as $\boldsymbol{\theta}^T \boldsymbol{f}$, which is a linear combination of features extracted from the generated trajectory ($\boldsymbol{f}$). $\boldsymbol{e}_t$ corresponds to the environmental parameters, including the road heading and the current and predicted future state of leading and following vehicles. $\boldsymbol{\theta}$ is the weight vector associated with the extracted features. The selected features and the constraints are demonstrated in the following sections.

2.2.1.2 Objective Function

By modifying the BSMs, specifically the speed and position data transmitted by the vehicle, the attack can generate falsified trajectory with an intentionally extended ETA compared to the ground truth. Besides, the attack also ensures that the generated falsified trajectory can mimic most of the normal driving behaviors. To achieve the above goals, 5 representative features are selected, capturing the aspects of normal car-following behavior, trajectory smoothness and the desired ETA.

1) $f_1 = \frac{1}{N}\sum_{t=1}^{N} a_t^2$. f_1 is the summation of vehicle's acceleration over the entire trajectory. f_1 ensures trajectory smoothness.
2) $f_2 = \frac{1}{N-1}\sum_{t=1}^{N-1}\left(\frac{\psi_{t+1}-\psi_t}{\Delta t}\right)^2$. f_2 denotes the average heading rate.
3) $f_3 = \frac{1}{N}\sum_{t=1}^{N}(\psi_{t+1} - \psi^{road})^2$. f_3 quantifies the difference between the vehicle heading and road heading. f_2, f_3 prevent the vehicle from deviating from the road.
4) $f_4 = \frac{1}{N}\sum_{t=1}^{N}\left(d_t^{des} - d_t\right)^2 = \frac{1}{N}\sum_{t=1}^{N}\left(v_t \times t_{headway} + d_s - d_t\right)^2$. f_4 is the difference between the desire and actual space headway with the leading vehicle.
5) $f_5 = (\frac{d_N^{stop}}{v_N} - ETA^{des})$. f_5 denotes the difference between the ETA at the end of the planning horizon and the desired ETA.

f_1, f_2, f_3 are designed to ensure the smoothness of the generated trajectory. f_4 captures the car following behavior. f_5 pushes the vehicle to reach the desired ETA by minimizing the difference between the estimated ETA and the desired ETA, calculated by **Equation 2**.

$$ETA^{des} = ETA^{init} + ETA^{offset} = \frac{d_0^{stop}}{v_0} + ETA^{offset} \quad (2)$$

2.2.1.3 Constraints

To generate a trajectory as realistic as possible, the optimization model is subject to vehicle safety constraints and the vehicle dynamics constraints. The vehicle safety constraint guarantees the minimum space headway between the vehicle under attack and the leading and following vehicle, increasing the difficulty for anomaly detection. Otherwise, the generated trajectory can easily be identified through cross validation. The vehicle dynamics constraints guarantee the generated trajectories follow the vehicle kinematic motion constraints and successfully pass the plausibility check. **Equations 3-4** demonstrate the vehicle safety constraints and **Equations 5-13** demonstrate the vehicle dynamic constraints.

$$(d_i^{follow\_travel} + d_s) - \delta_i^{follow} \times M \leq d_i^{travel}, i \in (1,2 \dots N) \quad (3)$$
$$d_i^{travel} \leq \left(d_i^{front_{travel}} - d_s\right) + \delta_i^{front} \times M, i \in (1,2 \dots N) \quad (4)$$

M is a constant large number. The binary parameters $\delta_i^{front}, \delta_i^{follow}$ indicate the presence of the leading and following vehicles. $d_i^{travel}, d_i^{front\_travel}$, and $d_i^{follow\_travel}$ denote the travel distance of the ego, leading and following vehicles at time step i. **Equations 3-4** ensure the minimum safety distance between the ego vehicle and the neighboring vehicles, avoiding the generated falsified vehicle trajectory exceeding the leading vehicle or overlapping with the following vehicle.

$$x_{t+1} = x_t + (v_t \Delta t + \frac{1}{2} a_t \Delta t^2) cos(\psi_t) \quad (5)$$
$$y_{t+1} = y_t + (v_t \Delta t + \frac{1}{2} a_t \Delta t^2) sin(\psi_t) \quad (6)$$
$$v_{t+1} = v_t + a_t \Delta t \quad (7)$$
$$\psi_{t+1} = \psi_t + \dot{\psi}_t \Delta t \quad (8)$$
$$v_t \leq v_{max} \quad (9)$$
$$a_{min} \leq a_t \leq a_{max} \quad (10)$$
$$\dot{\psi}_{min} \leq \dot{\psi}_t \leq \dot{\psi}_{max} \quad (11)$$
$$v_{stop} \leq v_t \quad (12)$$
$$\Delta\psi_{min} \leq \psi_t - \psi^{road} \leq \Delta\psi_{max} \quad (13)$$

Equations 5-8 demonstrate the vehicle kinematic motion constraints. **Equations 9-11** prevent the variables from exceeding kinematic boundaries, including speed limit, maximum acceleration/deceleration and maximum heading rate. **Equation 12** sets a minimum speed for the generated falsified trajectory, preventing the vehicle from being identified as a stopped vehicle. **Equation 13** guarantees the lane keeping of the vehicle.

2.2.1.4 Example Attack Trajectories

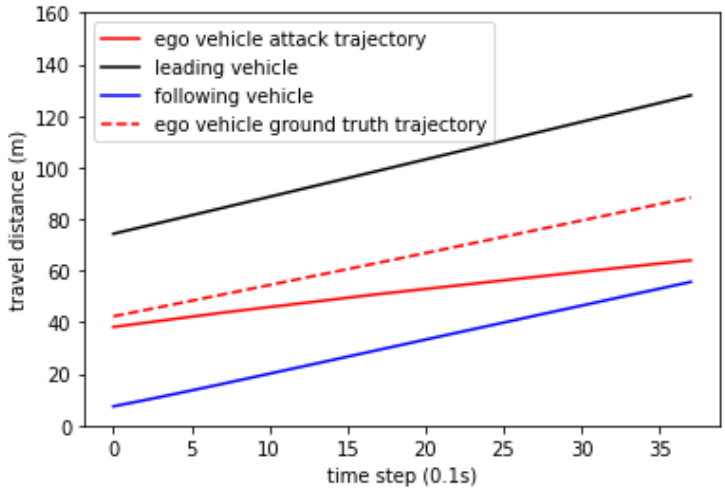

Figure 2: Ego vehicle, leading vehicle and following vehicle trajectories

Figure 2 demonstrates the time-space diagram of the ego, leading and following vehicles. The x-axis represents the time step, and the y-axis represents the traveled distance. The red curve denotes the falsified BSM trajectory broadcast by the ego vehicle under attack and the dashed red curve denotes the actual trajectory. The black and blue curves denote the trajectories of the leading and following vehicles. The figure shows that the falsified trajectory decelerates early, even when far from the leading vehicle, to achieve the ETA attack objective. Additionally, the attacked trajectory maintains safe distances from neighboring vehicles, making detection more challenging.

2.2.2 VASP Implementation

Estimated Time of Arrival (ETA) attack API is built on VASP. In the developed ETA attack API, the attack can access neighboring vehicle states (i.e., vehicle speed, acceleration, position) and traffic signal states (i.e., elapsed green time, current signal phase, signal plan update time) through SUMO interface. Given the neighboring vehicle status and signal state, an optimization problem is formulated to generate falsified vehicle trajectories, as described in **Section 2.2.1**. The generated falsified trajectories are distributed by the victim vehicle through VEINS interface.

Similar to the attacks provided by VASP, the ETA attack API enables users to adjust attack parameters such as malicious vehicle rate. Besides, the attacker can also customize the desired ETA of the victim vehicle to control the attack aggressiveness. Different desired ETA can affect the signal control systems in various ways, leading to varying vehicle delays at the intersection. In addition to customized attack parameters, the ETA attack API also supports vehicle trajectories collection, including both ego (attack) vehicles and neighboring vehicles. The collected trajectories can be used for analyzing and developing relative anomaly detection algorithms.

2.3 MSF Attack API

2.3.1 Modeling Multi-Sensor Fusion (MSF) Attack

2.3.1.1 MSF Attack Introduction

As demonstrated in **Section 2.1**, MSF attack has significant impact on transportation system safety and mobility. It is essential to model and evaluate the MSF attack. To evaluate the MSF attack, Shen et al. [7] utilized the LGSVL simulator, an autonomous vehicle simulator with Apollo 5.0 as the autonomous driving platform. The simulation incorporates the complete Baidu Apollo AD system, including hardware components such as GPS, IMU, camera and LiDAR, along with software modules for localization, prediction, planning and control. The entire simulation process is time-consuming and complicated, making it challenging to replicate and to integrate with other simulation platforms and transportation applications.

Since our main objective is to model the attack's impact on transportation application safety and mobility, which is primarily reflected by the falsified trajectories. Instead of replicating the complicated attack pipeline, we focus on modeling the trajectory-level attack behaviors. The MSF attack contains two stages, including the vulnerability profiling stage and the aggressive spoofing stage. In the vulnerability spoofing stage, the attacker adds constant deviation to the GPS channel and observes the localization result of the MSF algorithm to identify the vulnerable period. In the aggressive spoofing stage, exponentially growing deviation is added to mislead the MSF algorithm and deviate the vehicle from the road. By analyzing the falsified trajectories, we observe that the lateral deviation patterns in the two stages are different. In the vulnerability profiling stage, the lateral deviation follows a Gaussian distribution with its mean close to zero. In the aggressive spoofing stage, the lateral deviation patterns follow a family of exponential function, depending on driving scenarios. Consequently, we model the lateral deviations of the two stages separately, as demonstrated in the following sections.

2.3.1.2 Lateral Deviation Modeling

In the vulnerability profiling stage, falsified trajectories are modeled based on the lateral deviation from the ground truth and the attack duration. As shown in Figure 3, the lateral deviation follows Gaussian distribution with a mean of -0.02 and a standard deviation of 0.048. Most of the lateral deviations fall into the range of (-0.2m,0.2m), with the majority clustering near zeros.

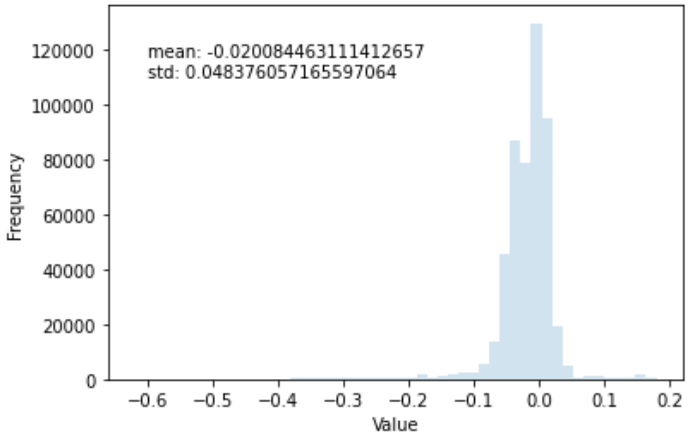

Figure 3: Vulnerability Profiling Stage Lateral Deviation Distribution

The duration of the vulnerability profiling stage is modeled using an exponential function, as shown in Figure 4. The fitted exponential function is represented in **Equation 14**, with the scale equals to 705.15. Since the lateral deviation in the vulnerability stage remains close to zero, this deviation only slightly shifts the vehicle from the lane center without crossing the lane boundary. The impact of the vulnerability profiling stage on neighboring vehicles is limited and can be ignored. Therefore, the error between the fitted curve and the data is considered acceptable.

$$f(x) = \frac{1}{scale} \times e^{-\frac{x}{scale}} \quad (14)$$

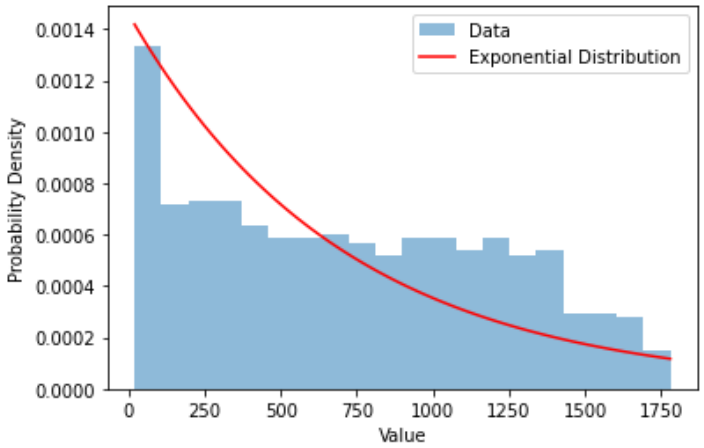

Figure 4: Vulnerability Profiling Stage Duration Distribution

The lateral deviation of the aggressive spoofing stage trajectory follows a family of exponential functions. **Equation 15** is applied to fit the curve. x represents the x_{th} attack point and $f(x)$ denotes the lateral deviation of the x_{th} point. a, b, c are function parameters. Figure 5 demonstrates the six lateral deviation patterns obtained from the original MSF attack, along with the corresponding fitted curves. The x-axis denotes time steps, and the y axis denotes the lateral deviation.

By analyzing trajectories collected from different driving scenarios, it is revealed that the lateral deviation pattern remains consistent for trajectories within the same scenario but varies among different driving scenarios, depending on the road geometry, speed limit and traffic volume. To account for these differences, a classification framework is proposed to distinguish between different driving scenarios and the corresponding lateral deviation patterns.

$$f(x) = c \times a^x + b \quad (15)$$

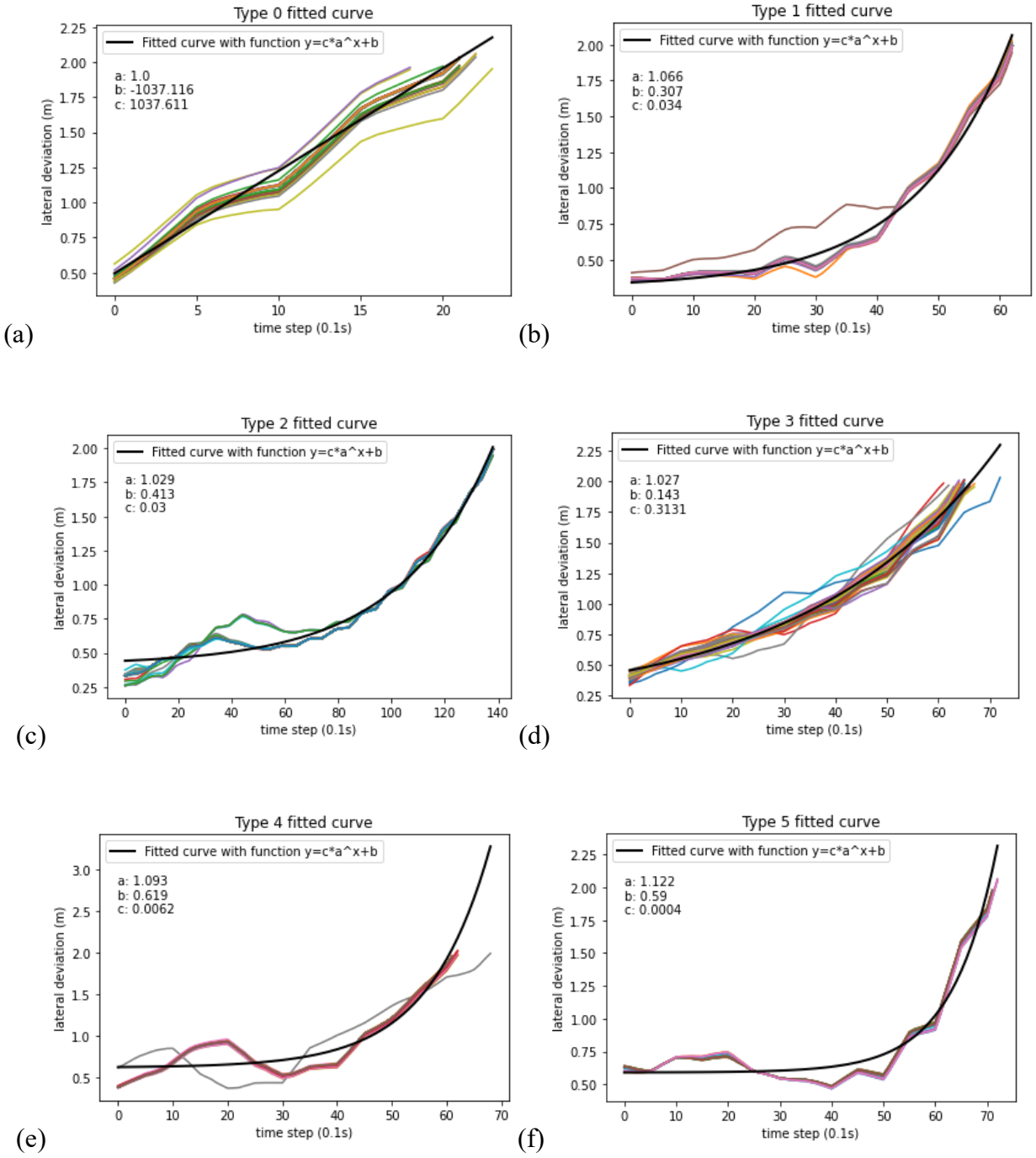

Figure 5: Lateral Deviation Profile and Fitted Exponential Curve

2.3.1.3 Trajectory Classification

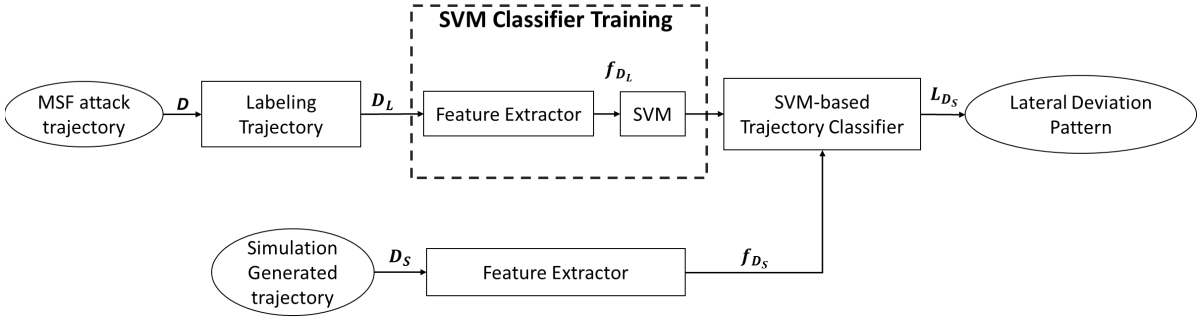

Figure 6: Trajectory Classification Framework

Figure 6 illustrates the trajectory classification framework. MSF attack trajectory dataset D_L is collected and labeled according to driving scenarios and the lateral deviation patterns. A feature extractor is applied to extract relevant features $\boldsymbol{f}_{D_L}$. SVM classifier is trained to distinguish different lateral deviation patterns given the extracted features. Given the proposed trajectory classification framework, a new attack trajectory can be generated from a simulation by identifying its lateral deviation pattern.

Feature extractor

Six features are designed to differentiate among various driving scenarios using trajectories from the vulnerability spoofing stage. The designed features focus on the distribution of vehicle kinematic parameters and can be easily obtained from the dataset.

1) Average speed: $f_1 = \frac{1}{N}\sum_i v_i$. N is the length of the vulnerability spoofing stage attack trajectory. v_i denotes the vehicle speed, f_1 calculates vehicle's average speed.
2) Maximum speed: $f_2 = \max_{i=1,2...N} v_i$. f_2 calculates the maximum speed in the vulnerability spoofing stage. f_1, f_2 reflect speed limit and traffic volume on the road.
3) Standard deviation of speed: $f_3 = \sqrt{\frac{\sum_i (v_i - \bar{v})^2}{(N-1)}}$. f_3 denotes the fluctuation of vehicle speed.
4) Average acceleration: $f_4 = \frac{1}{N}\sum_i a_i$. f_4 calculates the average acceleration.
5) Maximum acceleration: $f_5 = \max_{i=1,2...N} a_i$. f_5 calculates the maximum acceleration. f_4, f_5 reflect the smoothness of the trajectory.
6) Standard deviation of acceleration: $f_6 = \sqrt{\frac{\sum_i (a_i - \bar{a})^2}{(N-1)}}$. f_6 denotes the fluctuation of vehicle acceleration. f_6 reflects the traffic volume. In comparison to a free-flow driving scenario, vehicles on congested roads are more likely to adjust their speed and acceleration frequently to ensure safety and keep consistent with the surrounding traffic flow.

A greedy algorithm is applied to select the features that most influence trajectory categories. f_2 and f_5 are identified as the critical features for categorizing attack trajectories from different driving scenarios.

Classification Results

699 original attack trajectories generated from the MSF attack are used to validate the proposed trajectory classification algorithm. 80% of trajectories are used for training and 20% of trajectories are used for testing. The result demonstrates that the proposed algorithm achieves 100% accuracy in distinguishing different types of attack trajectories. Figure 7 denotes the distribution of selected features, with the x-axis representing maximum speed, and the y-axis representing maximum acceleration. The figure illustrates that the selected features can effectively discriminate among different types of attack trajectories.

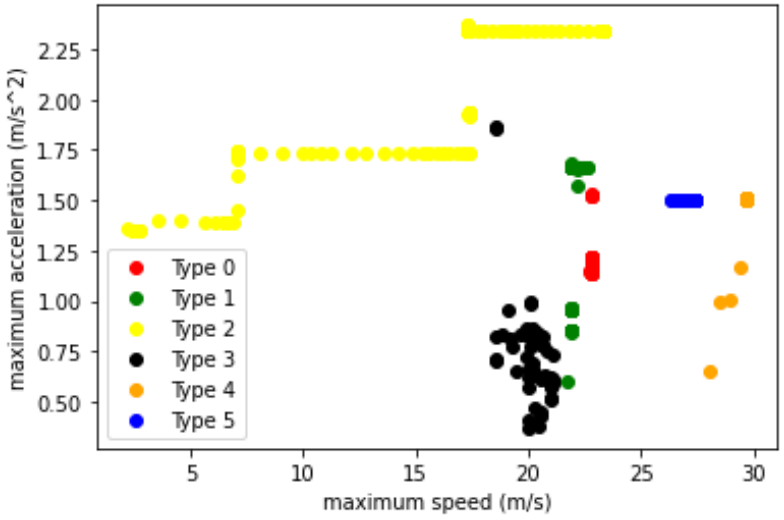

Figure 7: MSF attack trajectory feature distribution

2.3.2 MSF Attack API

The MSF attack API enables users to access the victim vehicle's state information (i.e., position and speed) through the SUMO interface. Based on these states, intelligent driver models predict the vehicle's future movement without attack. Lateral deviations are then generated for the two attack stages using the proposed surrogate model. These deviations are added to the predicted future position, and the attacker controls the victim vehicle's movement to the spoofed position using the SUMO interface.

This API also allows users to customize attack parameters, such as the malicious vehicle rate, as well as controlling traffic volume. Since the MSF attack directly influences the vehicle's movement, neighboring vehicles will react accordingly. Therefore, the safety impact of the attack can vary depending on the traffic volume. Additionally, the MSF attack API enables users to collect both falsified vehicles and neighboring vehicle trajectories. These trajectories can be analyzed to assess the impact of the MSF attack on transportation system safety and mobility, such as crash rate and vehicle delay.

CHAPTER 3

Route Guidance Attack API

3.1 Overview of Route Guidance Attacks (RGAs)

The increasing reliance on navigation systems in connected and autonomous vehicles (CAVs) has introduced new vulnerabilities in transportation systems, one of which is Route Guidance Attacks [17]. RGAs exploit the trust that travelers place in navigation systems by manipulating routing information, thereby redirecting vehicles onto suboptimal or inefficient routes. These attacks can have profound consequences on urban mobility, including increased congestion, extended travel times, and compromised traffic safety.

In RGAs, attackers alter routing information through various mechanisms, such as falsifying traffic conditions, introducing phantom congestion, or suggesting alternate routes that unnecessarily lengthen trip distances. These manipulations can be implemented dynamically, targeting specific locations and time periods to maximize disruption [18-20]. For instance, false reports of traffic congestion during peak hours can redirect significant traffic flow to alternate routes, creating artificial bottlenecks while leaving primary routes underutilized. The cumulative effect of such attacks on traffic dynamics can propagate throughout the network, leading to cascading congestion and delays.

RGAs are particularly concerning because of their scalability and the difficulty of detection. Attackers can target individual vehicles or large groups of users simultaneously, leveraging vulnerabilities in Vehicle-to-Everything (V2X) communication systems. As modern navigation systems become increasingly integrated with CAV technologies, understanding the mechanisms and impacts of RGAs is essential for ensuring the resilience of transportation networks.

3.2 Development of the RGA API

3.2.1 Objectives and Features

The RGA API is designed to model and analyze the impacts of GPS spoofing-based RGAs, where false GPS signals manipulate route information to redirect vehicles onto suboptimal or inefficient paths. This capability addresses significant gaps in existing simulation tools by providing a comprehensive framework for simulating and analyzing such attacks across transportation networks.

The primary objectives of the RGA API are to enable realistic GPS spoofing attacks and assess their network-level impacts. It quantifies attack impacts through metrics such as travel time deviations, route inefficiency, and congestion propagation. Furthermore, the API supports dynamic GPS spoofing scenarios by manipulating location data in real-time, enabling users to specify attack parameters such as geographic coverage, spoofing accuracy, intensity, and timing. Advanced analytics and visualization tools are integrated into the API to produce outputs like congestion propagation maps and trip delay histograms, enabling detailed evaluation of RGA outcomes.

3.2.2 RGA Algorithm

The RGA API is built around two modules: the spoofing engine and impact analysis module. These modules interact to generate and evaluate GPS spoofing-based RGAs. The spoofing engine generates GPS spoofing attacks by manipulating location data. It supports both static spoofing, where constant false locations are used, and dynamic spoofing, which introduces time-varying false locations. The engine can target individual vehicles, groups, or entire fleets, making it flexible for diverse attack scenarios. The simulation integrates the spoofing engine with Open Street Maps (OSMs). It facilitates updates to vehicle trajectories based on manipulated GPS signals. Finally, the impact analysis module evaluates the consequences of RGAs by traffic congestion patterns.

Below is the pseudocode for the RGA implementation:

Algorithm 1: GPS Spoofing-Based RGA

Input: City Name, Center Coordinates (latitude, longitude), Number of Trips (N), Number of Waypoints (W), Spoofing Parameters
Output: Map with Trajectories (Original and Spoofed), Trajectory Data CSV

1. Initialize the city map with OpenStreetMap road network for the specified city.
2. Generate N trips with W waypoints each:
   2.1. Select W random nodes from the road network as waypoints.
   2.2. For each pair of consecutive waypoints:
      2.2.1. Calculate the shortest path using the road network.
      2.2.2. Append coordinates of the path to the original route.
3. Randomly select a waypoint to apply GPS spoofing:
   3.1. Generate a spoofed GPS location by shifting the true location within a specified distance.
   3.2. Replace the true location with the spoofed location for subsequent routing.
4. Recalculate the shortest paths from the spoofed waypoint to subsequent waypoints:
   4.1. Adjust the trajectory to reflect the spoofed GPS data.
   4.2. Append coordinates of the new path to the spoofed route.
5. Visualize results:
   5.1. Add original and spoofed routes to the map with distinct styles (e.g., dashed lines for spoofed routes).
   5.2. Mark the attack point with a unique icon.
6. Save trajectory data to a CSV file, including trip index, route type (original/spoofed), and GPS coordinates.
7. Save the map as an HTML file.

3.2.3 Implementation of RGA

The RGA algorithm was implemented in Python, leveraging libraries such as Folium for visualization, OSMnx for downloading and manipulating road networks, and NetworkX for calculating shortest paths. Below are the key steps in the implementation:

1. Road Network Preparation: Road networks are downloaded for the specified city using OSMnx. Edge attributes, such as travel time and speed limits, are added to facilitate shortest path calculations.
2. Waypoint Generation: Random waypoints are selected from network nodes to create hypothetical trips.
3. Shortest Path Calculation: The shortest path between consecutive waypoints is calculated using Dijkstra's algorithm, considering edge weights like distance or travel time.
4. GPS Spoofing: A waypoint is randomly selected for GPS spoofing. The spoofed location is generated by introducing a drift in latitude and longitude, calculated based on a specified spoofing radius.
5. Route Recalculation: Shortest paths are recalculated from the spoofed location to subsequent waypoints, generating the spoofed route.
6. Visualization and Data Export: The map is updated with original and spoofed routes, attack points, and trajectory details. Trajectory data is exported to a CSV file for further analysis.

The implementation allows researchers to simulate and analyze the effects of GPS spoofing-based RGAs on traffic networks in a realistic and scalable manner, as shown in Figure 8.

a. Trajectories in New York City

b. Blue mark is a spoofed location, and red mark is the attacked location in New York City

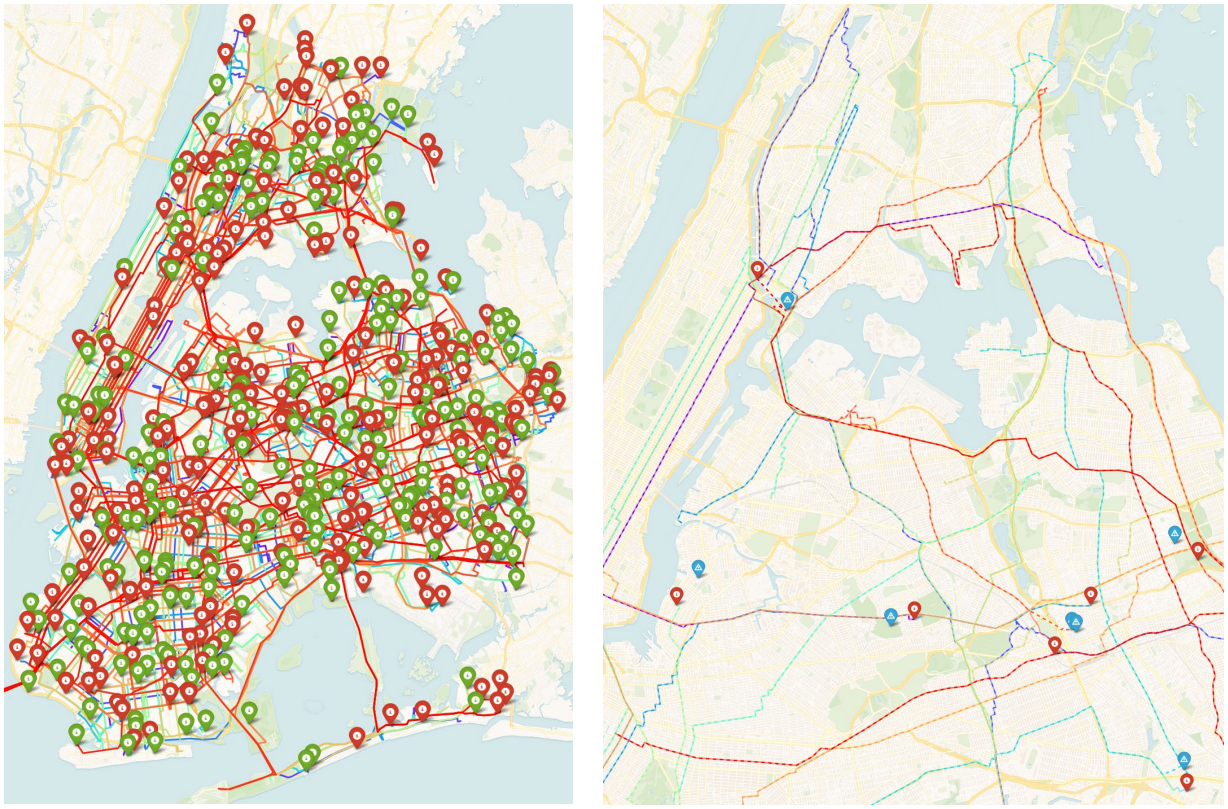

Figure 8 Example of Trajectories and RGAs in New York City

3.3 A Bounded Rational Route Guidance Attack Framework

3.3.1 Motivation and Objective

Modern navigation platforms fuse crowd sourced traffic data with shortest path algorithms to deliver real time guidance. While this architecture improves network efficiency, it also opens a new cyber physical threat surface. Earlier work on GPS spoofing and data poisoning shows that attackers can misreport link travel times and thereby redirect drivers, yet most studies either assume that users follow any recommendation blindly or examine attacks that are easy to detect. PHANTNAV (Phantom Navigation with bounded rational drivers) fills this gap by explicitly modeling a driver who is tolerant of small detours but rejects advice that appears implausible, that jointly (i) models the driver as a boundedly rational follower who accepts only modest travel-time deviations and (ii) learns stealthy yet disruptive travel-time falsifications through a Wasserstein Generative Adversarial Network with two critics. The attacker therefore optimizes a delicate balance: the forged congestion reports must stay within routine traffic variance to avoid anomaly flags, while still being large enough to push traffic onto inferior routes.

Figure 9 visualizes the fundamental trade off that guides PHANTNAV optimization. The orange curve shows that as the attacker raises intensity more drivers detect the anomaly, so the success rate falls. The blue curve shows that each compliant driver now experiences a larger individual delay. The product of these two effects is the green dashed curve, which forms an inverted U. The apex identifies the optimal intensity: it is moderate enough to remain believable yet strong enough to create significant detours. The purple marker denotes the point that PHANTNAV learns automatically through its dual critic training. The shaded bands indicate regions where intensity is either too mild to matter or so extreme that almost all users defect, both of which yield little aggregate harm.

When this conceptual curve is mapped onto empirical data, the peak shifts with congestion level. In free flow the optimum lies near the lower bound of indifference band because spare capacity dampens any excess delay. Under heavy demand the optimum moves rightward since even small additional traffic can destabilize queues, allowing the attacker to impose larger distortions without losing credibility.

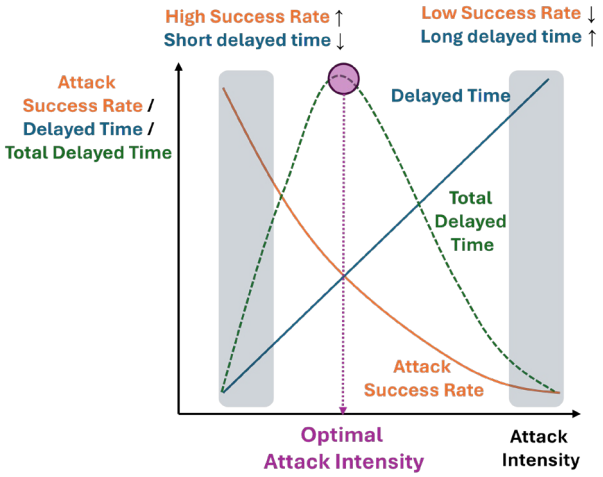

Figure 9: Relationship between attack intensity, attack success rate, delayed travel time per driver, and total delayed time

3.3.2 Implementation of PHANTNAV

At the beginning of each epoch the procedure samples a batch of origin–destination pairs and retrieves the corresponding ground-truth link costs $c = \{c_e\}$. The generator G_θ receives this cost vector together with a noise draw and outputs a perturbation Δc. A clipping step enforces the proportional bound $|\Delta c_e| \le \alpha c_e$, guaranteeing that every falsified cost remains within the prescribed attack intensity α. The perturbed cost vector $\tilde{c} = c + \Delta c$ is then combined with authentic samples to refresh two critic networks. The *standard critic* D_ϕ^{std} is updated using the Wasserstein distance with a gradient penalty so that it continues to distinguish statistically realistic costs from outliers. In parallel the algorithm solves a shortest-path problem under both $\tilde{c}$ and the baseline ccc to obtain the induced travel-time change ΔT; the *bounded-rational critic* D_ψ^{BR} is trained on this signal so that it rewards only those perturbations that lie inside the driver indifference band γ.

Algorithm 1 PHANTNAV

INPUT: Generator G_θ; Discriminators $D_\phi^{std}, D_\psi^{BR}$; attack intensity α; indifference band γ; max epochs N_{epoch}.
OUTPUT: Trained model $G_\theta, D_\phi^{std}, D_\psi^{BR}$.
procedure: PHANTNAV

1: **for** epoch $= 1$ to N_{epoch} **do**
2: **Critic Updates**:
3: (a) Sample O-D pairs, get true costs $c = \{c_e\}$.
4: (b) Generate $\Delta c = G_\theta(z, c)$, then clip to $[-\alpha c_e, \alpha c_e]$.
5: (c) Form $\tilde{c} = c + \Delta c$ (fake) and collect real samples.
6: (d) Update D_ϕ^{std} using WGAN objective.
7: (e) Find P^* under $\tilde{c}$, P^{opt} under c, obtain ΔT.
8: (f) Update D_ψ^{BR}.
9: **Generator Update**:
10: (a) Compute combined loss $L_G(\theta)$.
11: (b) Perform gradient descent on $\nabla_\theta L_G(\theta)$.
12: **end for**

After the critical updates, the generator performs a single gradient descent step against a combined objective that reflects both realism and behavioral acceptance. Specifically, the loss $L_G(\theta)$ aggregates the Wasserstein critic score, the bounded-rational critic score, and a penalty that is proportional to the negative of total additional travel time, thereby encouraging larger congestion effects while respecting the dual critic constraints. By repeating Steps 1 through 11 for N_{epoch} epochs, PHANTNAV converges to a generator G_θ that fabricates cost vectors that are indistinguishable from routine fluctuations yet still induce maximal aggregate delay within the bounds of driver tolerance. The final output of the algorithm consists of the trained generator and both critics, which together can be embedded in a real-time navigation pipeline to execute stealthy and effective route guidance attacks.

3.4 Network-Level Traffic Dynamics Model under RGAs

3.4.1 Generalized Bathtub Model (GBM)

The Generalized Bathtub Model (GBM) is a macroscopic traffic flow model that captures network-wide traffic dynamics by incorporating trip distance distributions and speed-density relationships [21]. It is particularly well-suited for analyzing the impacts of GPS spoofing-based RGAs, as it provides a framework for understanding how route manipulation affects traffic flow and congestion propagation across the entire network. The GBM models the number of active trips in the network, their remaining distances, and the overall traffic density through conservation laws. Key components include:

- Trip Distance Distributions: Represent the distribution of remaining distances for vehicles in the network, enabling analysis of how RGAs alter travel patterns.

- Speed-Density Relationship: Based on the Macroscopic Fundamental Diagram (MFD), this relationship links the average speed of vehicles to the overall network density, capturing the impact of congestion.
- Conservation Laws: These equations ensure the physical consistency of traffic flow. For example, the total number of trip miles in the network at any time is the sum of initial trip miles, trip miles added by inflows, and trip miles subtracted due to completed trips.

3.4.2 Extensions for GBM under RGAs

To analyze the impacts of GPS spoofing-based RGAs, the GBM is extended to include key features that capture manipulated routing information, congestion dynamics, and attack timing. These extensions highlight our contributions by introducing modifications to the trip distribution function, inflow dynamics, and congestion propagation models. Details are shown in [22].

To solve the extended GBM under GPS spoofing scenarios, finite numerical approaches are applied. This method discretizes the GBM equations, enabling real-time simulation of traffic dynamics under attack. Iterative adjustments account for spoofing effects, dynamically updating trip distance distributions and density-flow relationships at each time step. Below is the pseudocode for the finite numerical solution method's implementation:

Algorithm 2: Finite Numerical Solution Method for Extended GBM under RGAs

Input: Time, Range of Trip Distance, Number of Intervals J (time) and I (trip distance), Inflow Rate, Initial Conditions, Attack Time, Attack Intensity, Mean Shift of Trip Distance by RGAs
Output: Number of Active Trips, Number of Trips with Remaining Distances, Number of Outflows

1. Initialize the traffic flow settings.
   1.1. Set the initial state variables based on the given initial conditions.
   1.2. Divide the simulation time into J discrete time steps of equal size.
   1.3. Divide the range of trip distances into I bins of equal size
2. Iterate over Time Steps: For each time step:
   2.1. Update the travel speed based on the speed-density relationship and the number of active trips.
   2.2. Adjust the characteristic trip distance for the current time step using the updated speed.
   2.3. Update the number of active trips:
      - Before the attack: Compute the number of active trips considering trip.
      - After the attack: Modify the number of active trips using the attack parameters.
   2.4. Iterate over all trip distance bins:
      - Before the attack: Update the trip distance distribution to reflect normal travel behavior.
      - After the attack: Modify the trip distance distribution to account for manipulated routing.
   2.5. Calculate the cumulative outflows using the conservation of active trips.
3. End Time Loop: After all-time steps, output the state variables, including the number of active trips, trip distance distributions, and cumulative outflows.

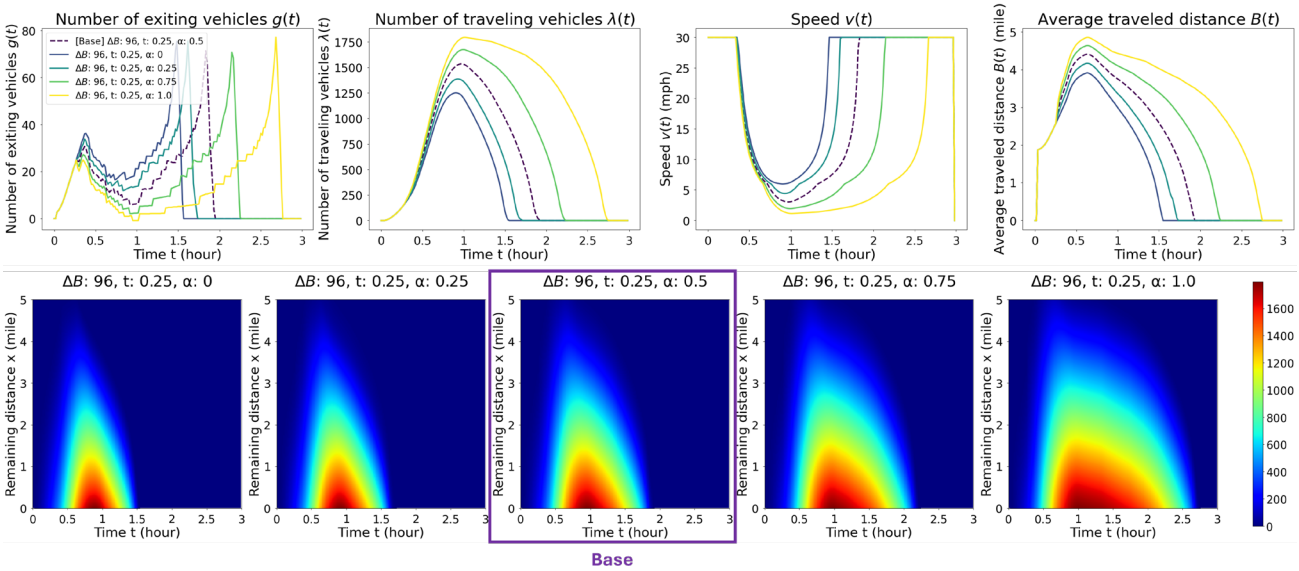

Figure 10: Examples of Network-Level Traffic Analysis: ΔB is an increase in travel distance, t is an attack time, and α is the rate of attacked vehicles in networks. Color means the number of vehicles.

CHAPTER 4

Federated Learning And Simulation Platform

Evaluating cybersecurity issues in transportation systems in real-world scenarios poses significant challenges. Current simulation tools have limitations that restrict their ability to fully replicate real-world conditions and effectively assess dynamic cyber threats. As part of this study, we initially conducted experiments related to cyber-attacks in distributed machine learning techniques. After implementing cyber-attacks during the FL training phase, we created a trained model and deployed it in an AV simulation platform to observe the vulnerabilities of cyber-attacks related to distributed ML techniques.

4.1 Introduction

Autonomous Vehicles (AVs), classified by the SAE from level 0 (no automation) to level 5 (full automation), rely on features like object recognition for safe navigation. However, integrating machine learning (ML) and deep learning (DL) in AVs introduces vulnerabilities to cyberattacks, particularly poisoning and inference attacks. Poisoning attacks manipulate data or model parameters to disrupt training, while inference attacks exploit aggregated updates to extract sensitive data. To address these challenges, federated learning (FL) has emerged as a privacy-preserving framework, keeping data local on edge devices while sharing only model updates, reducing privacy risks and communication costs.

Several studies have advanced FL-based security for AVs. BDFL (Chen et al. 2019) uses Byzantine-Fault-Tolerant techniques for secure model training, while privacy-aware FL [23] integrates blockchain and differential privacy for Vehicular Cyber-Physical Systems. OQFL [24] employs quantum optimization to defend against attacks, and SemBroc-RF [25] combines reinforcement learning with encrypted multi-party computation for enhanced security. GeFL [26] refines local models and encrypts edge data, improving accuracy and reducing data transmission. These methods underscore the potential of FL in securing AV systems against adversarial threats.

4.2 Methodology

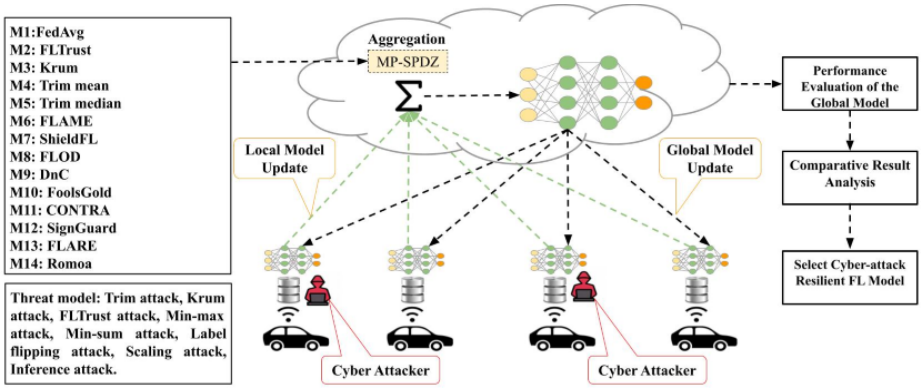

Figure 11: Overview of the empirical analysis of secure FL for AV applications.

Our study focuses on implementing secure FL aggregation algorithms and Secure MPC to address cyber-attacks in AV applications, particularly poisoning and inference attacks. Using the LISA traffic light dataset, we evaluated fourteen aggregation techniques, including FedAvg, FLTrust, Krum, Trim mean, Trim median, FLAME, ShieldFL, FLOD, DnC, FoolsGold, CONTRA, SignGuard, FLARE, and Romoa, under seven categories of attacks: Trim attack, Krum attack, FLTrust attack, Min-max attack, Min-sum attack, Label flipping attack, Scaling attack, and Inference attack. Inspired by SAFEFL [27], an MPC-based framework using PyTorch for training and MP-SPDZ [28] for secure aggregation, we tailored methodologies with hyperparameter optimization to examine the performance and resilience of these techniques during both training and inference phases. The empirical results highlighted the effectiveness of secure aggregation techniques in mitigating threats and provided a foundation for developing robust, attack-resilient FL strategies for AV applications.

In our threat model, poisoning attacks involve compromised devices manipulating training data or model gradients to mislead the global model, resulting in incorrect classifications. Inference attacks, on the other hand, aim to extract sensitive information by analyzing aggregated updates. Figure 1 presents an overview of our empirical study, where AVs act as edge clients with private datasets, refining the global model by sharing locally updated parameters. Our main goal is to develop resilient FL techniques capable of mitigating these cyber threats while ensuring robust AV traffic light detection in the presence of adversarial attacks.

4.3 Result Analysis

Table 1: Best performance in each attack (represented in each column of the table).

Attack / Aggregation	No	Trim	Krum	FLTrust	Min-Max	Min-Sum	Label flipping	Scaling
M1: FedAvg	99.74%	99.61%	**99.80%**	99.76%	97.97%	99.58%	99.65%	51.30%
M2: FLTrust	98.60%	98.71%	98.53%	98.56%	98.41%	98.65%	98.34%	98.18%
M3: Krum	98.38%	98.37%	84.31%	97.76%	98.52%	98.56%	98.61%	98.64%
M4: Trim mean	99.45%	98.26%	99.02%	98.65%	98.36%	98.92%	99.67%	99.44%
M5: Trim Median	99.17%	98.00%	98.92%	98.11%	98.07%	98.66%	99.39%	98.99%
M6: FLAME	99.41%	99.39%	99.26%	99.18%	99.27%	99.05%	99.52%	99.44%
M7: ShieldFL	81.44%	79.15%	02.92%	43.78%	74.61%	82.27%	73.03%	66.22%
M8: FLOD	**99.91%**	**99.72%**	99.63%	**99.97%**	96.25%	**100.0%**	98.50%	99.85%
M9: DnC	99.53%	99.56%	99.32%	99.00%	99.32%	99.06%	99.73%	99.62%
M10: FoolsGold	51.20%	45.88%	45.88%	45.88%	51.20%	96.60%	51.20%	97.16%
M11: CONTRA	51.20%	51.20%	02.92%	00.00%	51.20%	46.14%	51.19%	51.20%
M12: SignGuard	99.67%	99.07%	99.45%	99.37%	**99.73%**	99.39%	**99.74%**	**99.73%**
M13: FLARE	98.87%	98.23%	98.87%	98.89%	96.03%	98.96%	99.57%	96.62%
M14: Romoa	99.21%	98.79%	99.47%	99.24%	98.79%	99.33%	99.63%	99.26%

Table 1 highlights the performance of secure FL techniques under adversarial attacks, showing accuracy after 200 iterations. FLOD achieves the highest accuracy (99.91%) without attacks, followed by FedAvg (99.74%), SignGuard (99.67%), and DnC (99.53%). Techniques like Trim Mean (99.45%) and FLAME (99.41%) also perform well, while ShieldFL (81.44%) and

FoolsGold/CONTRA (51.20%) lag significantly. FLOD proves highly robust against Trim (99.72%), FLTrust (99.97%), and Min-Sum (100%) attacks. FedAvg excels in Krum attacks (99.80%), and SignGuard performs well against Min-Max, Label Flipping, and Scaling attacks. FLAME, FLTrust, and DnC also show strong resilience, with minimal accuracy deviations under attacks. However, methods like ShieldFL and CONTRA are severely compromised by Krum and FLTrust attacks, while other techniques demonstrate moderate security against specific threat models.

4.4. Simulation with Carla Simulation Platform

Following the empirical analysis, we transitioned to leveraging the CARLA simulation platform to evaluate the real-world applicability of FL-trained models and the impact of adversarial attacks on autonomous vehicle (AV) systems. Initially, a machine learning model was trained for object detection using real-time traffic data and deployed into the CARLA environment. Within this setup, we simulated various adversarial attacks, such as camera blinding and backdoor attacks, to assess their impact on AV behavior, safety, and decision-making during object detection tasks. Backdoor attacks were particularly examined to understand how manipulated training data, such as misleading pixels in input images, could mislead traffic light detection algorithms. These attacks exploit the iterative nature of FL by introducing poisoned local updates that propagate into the globally aggregated model, affecting the entire connected system of AVs. This highlighted the critical vulnerabilities of FL models in AV applications and emphasized the importance of robust defenses.

Building on related works like Simutack [29], which focus on simulating sensor attacks in CARLA, our project utilized CARLA's capabilities for generating realistic sensor data to further analyze the dynamic impact of adversarial attacks. For the initial implementation, a trained YOLOv3 object detection model was deployed to detect objects using real-time AV camera data. Specific attack scenarios, such as manipulating AV camera sensor data to simulate blinding attacks, were initiated to evaluate the robustness of object detection under adversarial conditions. These simulations revealed critical vulnerabilities in AV systems, emphasizing the need for robust defense mechanisms. This work sets the foundation for future investigations into defensive techniques, including their application during the training or inference phases, to enhance the security and reliability of AV systems in adversarial environments.

CHAPTER 5

Conclusions

In this project, we developed various cyber attack APIs for individual vehicle sensors, vehicle and infrastructure operations, and network-level applications. The developed APIs are implemented in multiple simulation environments.

At the vehicle level, the project focused on implementing secure FL aggregation algorithms and Secure MPC to address cyber-attacks in AV applications, particularly poisoning and inference attacks. Using the LISA traffic light dataset, we evaluated fourteen aggregation techniques under seven categories of sensor spoofing/poisoning attacks. Following the empirical analysis, we transitioned to leveraging the CARLA simulation platform to evaluate the real-world applicability of FL-trained models and the impact of adversarial attacks on autonomous vehicle (AV) systems.

At the traffic level, two typical data spoofing attacks were selected, including the estimated time of arrival (ETA) attack and the multi-sensor fusion (MSF) attack, and modeled and integrated on VASP, a V2X simulation platform developed by Qualcomm. The selected attacks target deteriorating mobility (ETA attack) and safety (MSF attack).

At the network level, a bounded rational route guidance attack (RGA) framework was proposed and an RGA API was built around two modules: the spoofing engine and impact analysis module. These modules interact to generate and evaluate GPS spoofing-based RGAs. The Generalized Bathtub Model (GBM) was used to model traffic flow and integrated with the RGA algorithm.

REFERENCES

[1] Lopez, P. A., Behrisch, M., Bieker-Walz, L., Erdmann, J., Flötteröd, Y. P., Hilbrich, R., ... and Wießner, E. (2018, November). Microscopic traffic simulation using sumo. In *2018 21st international conference on intelligent transportation systems (ITSC)* (pp. 2575-2582). IEEE

[2] D. Cui, Y. Shen, H. Yang, Z. Huang, K. Rush, P. Huang, and P. Bujanovic, "Extensible co-simulation framework for supporting cooperative driving automation research," *Transportation research record*, vol. 2677, no. 3, pp. 1067–1079, 2023.

[3] M. R. Ansari, J. Petit, J.-P. Monteuuis, and C. Chen, "Vasp: V2x application spoofing platform," in 2023 Vehicle Security Symposium, *NDSS*, 2023.

[4] Narain S, Ranganathan A, Noubir G. Security of GPS/INS Based On-road Location Tracking Systems. In: *2019 IEEE Symposium on Security and Privacy (SP)*. 2019. p. 587–601.

[5] Kamal M, Barua A, Vitale C, Laoudias C, Ellinas G. GPS Location Spoofing Attack Detection for Enhancing the Security of Autonomous Vehicles. In: *2021 IEEE 94th Vehicular Technology Conference (VTC2021-Fall)*. 2021. p. 1–7.

[6] Cao, Y., Xiao, C., Cyr, B., Zhou, Y., Park, W., Rampazzi, S., ... and Mao, Z. M. (2019, November). Adversarial sensor attack on lidar-based perception in autonomous driving. In *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security* (pp. 2267-2281).

[7] Shen, J., Won, J. Y., Chen, Z., and Chen, Q. A. (2020). Drift with devil: Security of {Multi-Sensor} fusion based localization in {High-Level} autonomous driving under {GPS} spoofing. In *29th USENIX security symposium (USENIX Security 20)* (pp. 931-948).

[8] Singh, P. K., Tabjul, G. S., Imran, M., Nandi, S. K., and Nandi, S. (2018, October). Impact of security attacks on cooperative driving use case: CACC platooning. In *TENCON 2018-2018 IEEE Region 10 Conference* (pp. 0138-0143). IEEE.

[9] Dadras, S., Gerdes, R. M., and Sharma, R. (2015, April). Vehicular platooning in an adversarial environment. In *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security* (pp. 167-178).

[10] Chen, Q. A., Yin, Y., Feng, Y., Mao, Z. M., and Liu, H. X. (2018, February). Exposing Congestion Attack on Emerging Connected Vehicle based Traffic Signal Control. In *NDSS*.

[11] Huang, S. E., Wong, W., Feng, Y., Chen, Q. A., Mao, Z. M., and Liu, H. X. (2020). Impact evaluation of falsified data attacks on connected vehicle based traffic signal control. *arXiv preprint arXiv:2010.04753*.

[12] Wan, G., Yang, X., Cai, R., Li, H., Zhou, Y., Wang, H., and Song, S. (2018, May). Robust and precise vehicle localization based on multi-sensor fusion in diverse city scenes. In *2018 IEEE international conference on robotics and automation (ICRA)* (pp. 4670-4677). IEEE.

[13] Li, Q., Queralta, J. P., Gia, T. N., Zou, Z., and Westerlund, T. (2020). Multi-sensor fusion for navigation and mapping in autonomous vehicles: Accurate localization in urban environments. *Unmanned Systems*, *8*(03), 229-237.

[14] Ansari, M. R., Petit, J., Monteuuis, J. P., and Chen, C. (2023, February). Vasp: V2x application spoofing platform. In *Proceedings Inaugural International Symposium on Vehicle Security and Privacy, ndss-symposium*.

[15] Sommer, C., German, R., and Dressler, F. (2010). Bidirectionally coupled network and road traffic simulation for improved IVC analysis. *IEEE Transactions on mobile computing*, *10*(1), 3-15.

[16] Varga, A., and Hornig, R. (2008, March). An overview of the OMNeT++ simulation environment. In *Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems and workshops* (pp. 1-10).

[17] La Fontaine, S., Muralidhar, N., Clifford, M., Eliassi-Rad, T., and Nita-Rotaru, C. (2022, June). Alternative route-based attacks in metropolitan traffic systems. In *2022 52nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)* (pp. 20-27). IEEE.

[18] Zeng, K. C., Shu, Y., Liu, S., Dou, Y., and Yang, Y. (2017, February). A practical GPS location spoofing attack in road navigation scenario. In *Proceedings of the 18th international workshop on mobile computing systems and applications* (pp. 85-90).

[19] Zeng, K. C., Liu, S., Shu, Y., Wang, D., Li, H., Dou, Y., ... and Yang, Y. (2018). All your {GPS} are belong to us: Towards stealthy manipulation of road navigation systems. In *27th USENIX security symposium (USENIX security 18)* (pp. 1527-1544).

[20] Cao, Y., Luo, Q., and Liu, J. (2019, May). Road navigation system attacks: a case on GPS navigation map. In *ICC 2019-2019 IEEE International Conference on Communications (ICC)* (pp. 1-5). IEEE.

[21] Jin, W. L. (2020). Generalized bathtub model of network trip flows. *Transportation Research Part B: Methodological*, *136*, 138-157.

[22] Ka, E., and Ukkusuri, S. V. (2025). Analytical Framework for Network-Level Traffic Flow Under Route Guidance Attacks: An Extension of the Generalized Bathtub Model. Paper presented at the 104th Annual Meeting of the Transportation Research Board, Washington, DC.

[23] Olowononi, F. O., Rawat, D. B., and Liu, C. (2021, January). Federated learning with differential privacy for resilient vehicular cyber physical systems. In *2021 IEEE 18th Annual Consumer Communications and Networking Conference (CCNC)* (pp. 1-5). IEEE.

[24] Yamany, W., Moustafa, N., and Turnbull, B. (2021). OQFL: An optimized quantum-based federated learning framework for defending against adversarial attacks in intelligent transportation systems. *IEEE Transactions on Intelligent Transportation Systems*, *24*(1), 893-903.

[25] Zhu, R., Li, M., Yin, J., Sun, L., and Liu, H. (2023). Enhanced federated learning for edge data security in intelligent transportation systems. *IEEE Transactions on Intelligent Transportation Systems*, *24*(11), 13396-13408.

[26] Parekh, R., Patel, N., Gupta, R., Jadav, N. K., Tanwar, S., Alharbi, A., ... and Raboaca, M. S. (2023). GeFL: Gradient encryption-aided privacy preserved federated learning for autonomous vehicles. *IEEE Access*, *11*, 1825-1839.

[27] Gehlhar, T., Marx, F., Schneider, T., Suresh, A., Wehrle, T., and Yalame, H. (2023, May). SafeFL: MPC-friendly framework for private and robust federated learning. In *2023 IEEE Security and Privacy Workshops (SPW)* (pp. 69-76). IEEE.

[28] Keller, M. (2020, October). MP-SPDZ: A versatile framework for multi-party computation. In *Proceedings of the 2020 ACM SIGSAC conference on computer and communications security* (pp. 1575-1590).

[29] Finkenzeller, A., Mathur, A., Lauinger, J., Hamad, M., and Steinhorst, S. (2023, June). Simutack-an attack simulation framework for connected and autonomous vehicles. In *2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring)* (pp. 1-7). IEEE.