Cyber Security of Traffic Signal Control Systems with Connected Vehicles

by

Shihong Ed Huang

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Civil and Environmental Engineering)
in the University of Michigan
2020

Shihong Ed Huang
edhuang@umich.edu
ORCID iD: 0000-0002-5583-7655

D E D I C A T I O N

I dedicate this dissertation to the following people:

To my beloved wife, my best friend, and my soulmate Landy Di Lu, without whom I could not finish this journey.

To my parents Dafeng Huang and Xiaoli Xu, who always support me.

ACKNOWLEDGMENTS

I would like to express my heartfelt thanks to my advisor Prof. Henry Liu, who has supported me over the past years. Prof. Liu could always identify key problems in research studies. The discussion with Prof. Liu greatly inspired me on my dissertation research. It was my pleasure and honor to work with Prof. Liu. This dissertation could not be completed without Prof. Liu.

I would also like to thank Prof. Morley Mao, Prof. Neda Masoud, and Prof. Yafeng Yin, who served on my dissertation committee and provided valuable comments and suggestions to my dissertation. It is a blessing to work with them.

Very special thanks to Dr. Yiheng Feng. Dr. Feng provided lots of guidance on my dissertation research and greatly assisted my research. Dr. Feng also provided mental support to me. I deeply appreciate all the help and encouragement from Dr. Feng.

I would like to thank my previous and current lab mates, particularly, Dr. Jie Sun, Prof. Xuan (Sharon) Di, Dr. Heng Hu, Dr. Jianfeng Zheng, Mr. Shengyin Shen, Dr. Yan Zhao, Dr. Lin Xiao, Dr. Weili Sun, Dr. Wai Wong, Mr. Zhen Yang, Dr. Shuo Feng, Mr. Xingmin Wang, Mr. Xintao Yan, Mr. Haowei Sun, and Ms. Lin Liu. I enjoyed studying, working, and playing with them.

I would like to thank other people that I have worked and collaborated with, particularly, Prof. Qi Alfred Chen, Mr. Shengtuo Hu, and Mr. Qingzhao Zhang.

Finally, I would also like to thank my friend Mr. Jim Carlson for hosting me at his house and for teaching me the wisdom of life.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Our world is becoming increasingly connected through smart technologies. The same trend is emerging in transportation systems, wherein connected vehicles (CVs) and transportation infrastructure are being connected through advanced wireless communication technologies. CVs have great potential to improve a variety of mobility applications, including traffic signal control (TSC), a critical component in urban traffic operations. CV-based TSC (CV-TSC) systems use trajectory data to make more informed control decisions, therefore can accommodate real-time traffic fluctuations more efficiently. However, vehicle-infrastructure connectivity opens new doors to potential cyber attacks. Malicious attackers can potentially send falsified trajectory data to CV-TSC systems and influence signal control decisions. The benefit of CV-TSC systems can be realized only if the systems are secure in cyberspace. Although many CV-TSC systems have been developed within the past decade, few consider cyber security in their system design. It remains unclear exactly how vulnerable CV-TSC systems are, how cyber attacks may be perpetrated, and how engineers can mitigate cyber attacks and protect CV-TSC systems. Therefore, this dissertation aims to systematically understand the cyber security problems facing CV-TSC systems under falsified data attacks and provide a countermeasure to safeguard CV-TSC systems. These objectives are accomplished through four studies.

The first study evaluates the effects of falsified data attacks on TSC systems. Two

TSC systems are considered: a conventional actuated TSC system and an adaptive CV-TSC system. Falsified data attacks are assumed to change the input data to these systems and therefore influence control decisions. Numerical examples show that both systems are vulnerable to falsified data attacks.

The second study investigates how falsified data attacks may be perpetrated in a realistic setting. Different from prior research, this study considers a more realistic but challenging black-box attack scenario, in which the signal control model is unavailable to the attacker. Under this constraint, the attacker has to learn the signal control model using a surrogate model. The surrogate model predicts signal timing plans based on critical traffic features extracted from CV data. The attacker can generate falsified CV data (i.e., falsified vehicle trajectories) to alter the values of critical traffic features and thus influence signal control decisions.

In the third study, a data-driven method is proposed to protect CV-TSC systems from falsified data attacks. Falsified trajectories are behaviorally distinct from normal trajectories because they must accomplish a certain attack goal; thus, the problem of identifying falsified trajectories is considered an abnormal trajectory identification problem. A trajectory-embedding model is developed to generate vector representations of trajectory data. The similarity (distance) between each pair of trajectories can be computed based on these vector representations. Hierarchical clustering is then applied to identify abnormal (i.e., falsified) trajectories.

In the final study, a testing platform is built upon a virtual traffic simulator and real-world transportation infrastructure in Mcity. The testing platform integrates the attack study and defense study in a unified framework and is used to evaluate the real-world impact of cyber attacks on CV-TSC systems and the effectiveness of defense strategies.

CHAPTER 1

Introduction

1.1 Motivation

We are moving into the era of the Internet of Things, in which nearly all physical devices will be connected and exchanging data. The same trend is emerging in transportation systems, wherein vehicles and infrastructure are being connected through advanced wireless communication technologies, such as Dedicated Short-Range Communications (DSRC) (Kenney, 2011) and cellular networks (Wang et al., 2017). These vehicles are called connected vehicles (CVs), which are capable of communicating with other CVs and transportation infrastructure via Vehicle-to-Vehicle (V2V) communication and Vehicle-to-Infrastructure (V2I) communication. Research has shown that CVs have great potential to improve traffic signal control (TSC). CV-based TSC (CV-TSC) systems are deployed in transportation infrastructure and use CV data (i.e., vehicle trajectories) to make more informed control decisions. Compared with conventional TSC systems that use data from fixed-location sensors, CV-TSC systems accommodate real-time traffic fluctuations more efficiently. However, vehicle-infrastructure connectivity also opens new doors to potential cyber attacks. Malicious attackers can potentially launch cyber attacks against CV-TSC systems to jeopardize these systems' operation. The benefits of CV-TSC systems can be realized only if the systems are secure in cyberspace. Although many CV-TSC systems have been developed within the past decade, few consider cyber security in their system design. It remains unclear exactly how vulnerable CV-TSC systems are, how cyber attacks may be perpetrated, and how engineers can mitigate cyber attacks and protect CV-TSC systems.

This dissertation aims to understand the cyber security problems facing TSC systems with CVs. First, an empirical study is presented to evaluate the effects of potential cyber attacks on different TSC systems. Based on the results from the empirical study, a formal attack study is then conducted in a realistic setting. Next, the focus of this dissertation shifts to exploring an effective defense strategy to safeguard CV-TSC systems. Finally, a testing platform is built upon a virtual traffic simulator and real-world transportation infrastructure in Mcity. The testing platform integrates the attack study and defense study in a unified framework and can be used to perform various attack and defense experiments.

More background information will be introduced in the next section, including an introduction to CVs and TSC systems, cyber security issues associated with TSC systems, and the threat model. The dissertation framework is then proposed, followed by the contributions of this dissertation. Finally, a summary of each chapter is provided.

1.2 Background

1.2.1 Connected Vehicles and Traffic Signal Control

CV technologies enable vehicles and transportation infrastructure to communicate with each other wirelessly. This communication mechanism ensures information exchange between CV users and therefore improves the safety and mobility of ground transportation. Given these benefits, the CV industry has witnessed substantial progress in development and deployment throughout the past decade. In 2011, the U.S. Department of Transportation (USDOT) initiated the Safety Pilot Model Deployment project to validate DSRC technology for V2V and V2I safety applications (Bezzina and Sayer, 2014). USDOT later launched three more CV pilot programs in New York City, Tampa, and Wyoming in 2016 as part of a national effort to test CV-based applications (USDOT, 2019a).

CVs have shown promise in enhancing various mobility applications, including TSC, a critical component in urban traffic operations. A typical CV-TSC system is illustrated in Figure 1.1. Each CV is equipped with an On-Board Unit (OBU), which broadcasts Basic Safety Messages (BSMs). A BSM records a CV's information including its location, speed, heading, and acceleration. Consecutive BSMs represent the vehicle trajectory. On the infrastructure side, an intersection is equipped with a Roadside Unit (RSU), signal controller, and traffic signals. The RSU uses received BSMs (i.e., trajectories) to optimize signal timing plans. The signal controller executes optimal signal timing plans and controls traffic signals to display corresponding colors. The signal controller is connected to a transportation management center, which can send commands remotely to the signal controller (e.g., time-of-day signal timing plans). Meanwhile, the RSU continuously broadcasts Signal Phase and Timing (SPaT) messages, which record current signal status (i.e., green/yellow/red) and remaining time. Based on continuously received BSMs, the CV-TSC system generates optimal signal timing plans and responds to real-time traffic demands.

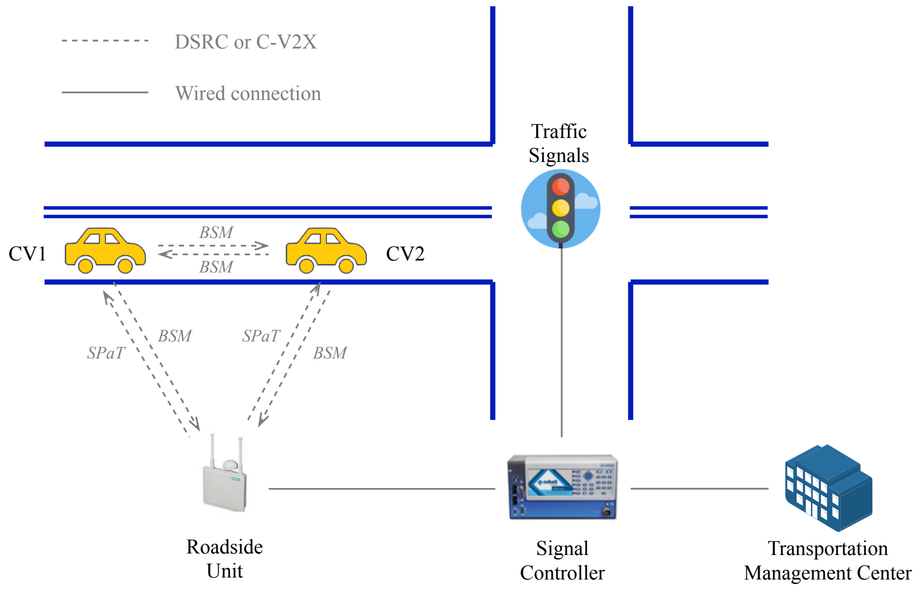


Figure 1.1: Illustration of a connected vehicle based traffic signal control system

Many studies have shown that CV-TSC systems often outperform conventional TSC systems in terms of cost and performance. Conventional TSC systems mainly rely on fixed-location sensors (e.g., loop detectors and cameras) for data collection and decision making. It is costly to install and maintain these fixed-location sensors as one intersection normally requires a suite of sensors. However, CV-TSC systems only need a single RSU for each intersection. This arrangement greatly reduces the associated costs. In addition, CV-TSC systems tend to perform better than conventional TSC systems because CV data provide temporal and spatial measurements of traffic and thus have higher resolution and quality than fixed-location sensors. Over the past decade, scholars have proposed numerous CV-TSC control models that outperform conventional TSC systems (Priemer and Friedrich, 2009; He et al., 2012; Goodall et al., 2013; Lee et al., 2013; Pandit et al., 2013; He et al., 2014; Guler et al., 2014; Feng et al., 2015; Beak et al., 2017; Li and Ban, 2018; Feng et al., 2018c; Zheng et al., 2018; Yu et al., 2018; Feng et al., 2018a; Yu et al., 2019; Yang et al., 2019).

In light of these benefits, it is believed that CV-TSC systems will eventually replace conventional TSC systems and become the next-generation TSC system (USDOT, 2019b). However, the complete implementation of CV technologies in transportation systems could take years or even decades. The transition from conventional TSC systems to CV-TSC systems may be similarly gradual. During this transition period, CV-TSC systems will operate under low CV market penetration. A recently developed representative CV-TSC system is the Multi-Modal Intelligent Traffic Signal System (MMITSS), which is a part of the Cooperative Transportation Systems Pooled Fund Study that focuses on CV Dynamic Mobility Applications. The goal of this project is to develop and field test CV-TSC systems that serve multimodal travelers including passenger vehicles, transit vehicles, emergency vehicles, freight vehicles, and pedestrians. The project consists of four main components: Intelligent Traffic Signal System (I-SIG), Signal Priority

(Transit-TSP, Freight-FSP, Emergency Vehicle-EVP), Mobile Accessible Pedestrian Signal System (PED-SIG) and a real-time performance observer (PERF-OBS). I-SIG provides real-time adaptive signal control services to general vehicles and is the CV-TSC system being evaluated in this dissertation. For descriptions of the other components of MMITSS, readers can refer to University of Arizona et al. (2016).

1.2.2 Cyber Security of Traffic Signal Control Systems

Growing cyber attacks have been reported against traffic control systems over the past several years. For instance, an Argentinian security expert hacked into New York City's wireless vehicle detection system via a cheap wireless device (Prigg, 2014). The loophole he discovered in the system enabled anyone to take complete control of vehicle detection devices and send fake data to traffic control systems. Although traffic signals were not controlled directly, fake vehicle data could lead to severe traffic congestion and higher crash risks. Another example emerged in Austin, Texas, where a variable message sign was altered to display "Zombies Ahead" rather than correct traffic information (Miller, 2009). This incident did not result in damage, but such mischief revealed potential security issues in transportation systems.

As indicated, TSC systems are undoubtedly susceptible to potential cyber attacks. It is important to understand the vulnerability of these systems by evaluating the impact of cyber attacks (i.e., extra travel time or delay caused by cyber attacks). Research on TSC-focused cyber attacks can be divided into two categories based on the attack mechanism.

The first research line assumes that attackers can gain access to TSC systems, either physically or remotely, to directly manipulate signal control decisions. For instance, Laszka et al. (2016) studied the vulnerability of transportation networks with fixed signal control. The author assumed that an attacker could compromise traffic signal controllers and arbitrarily change signal timing plans. Ernst and Michaels (2017) performed simulations to numerically evaluate the potential impact of an attack by assuming that an attacker could alter the synchronization of traffic signals on a corridor. Perrine et al. (2019) treated attacked traffic signals as stop signs and quantified the possible effects of cyber attacks. Ganin et al. (2019) modeled cyber attacks as speed reduction for attacked links in the traffic network. In addition to urban TSC systems, Reilly et al. (2016) focused on freeway TSC systems and assumed an attacker could compromise ramp meters and therefore control the on-ramp traffic flow to the freeway.

The second research stream posits that attackers cannot directly manipulate signal control decisions, but can send falsified data to influence these decisions. Ghafouri et al. (2016) assumed that sensor data could be used to generate fixed signal timing plans and that an attacker could tamper with sensor data, resulting in inefficient traffic signal scheduling. Chen et al. (2018) analyzed I-SIG and found that an attacker could generate falsified CV data to fool I-SIG by manipulating either arrival time or queue length. Yen et al. (2018) analyzed different backpressure-based scheduling algorithms and discovered that falsified arrival data could influence the signal scheduling of a TSC system. In addition to influencing signal control decisions, Jeske (2013) and Sinai et al. (2014) found that by creating ghost drivers and reporting fake GPS coordinates to social navigation systems, attackers could generate virtual traffic jams to influence routing decisions in the social navigation systems (e.g., Google Maps or Waze).

Despite their revelations, studies related to cyber security problems can suffer from two major drawbacks. First, such work (Laszka et al., 2016; Reilly et al., 2016; Ernst and Michaels, 2017; Chen et al., 2018; Yen et al., 2018; Ganin et al., 2019; Perrine et al., 2019) typically adopts a white-box attack scenario, the assumption of which lacks feasibility. This white-box attack scenario assumes that attackers have full access to the TSC system and/or control model and can freely manipulate traffic signal phasing and timing. For instance, in a study of a conventional TSC system, Perrine et al. (2019) presumed that traffic signals could be selectively disabled to flashing-red status (equivalent to a four-way stop-sign intersection). Yet the study did not provide justification for how this action could be achieved. In a study on I-SIG, Chen et al. (2018) assumed that the signal control model's source code would be known to an attacker, in which case a comprehensive security analysis of I-SIG could be performed. However, in a real-world scenario, the likelihood of an attacker gaining access to the source code is low. Although the white-box approach can provide the upper bound of system impacts under cyber attacks, the white-box assumption itself is strong and unrealistic. Second, most studies have merely examined the impact of cyber attacks without delineating concrete defense strategies. More work is therefore needed to investigate possible defense strategies to protect TSC systems.

Overall, although prior studies have provided valuable insights into the influences of cyber attacks against CV-TSC systems, a realistic and systematic study of this topic remains lacking.

1.2.3 Threat Model

Due to the unique features of CV-TSC systems (i.e., connectivity between CVs and transportation infrastructure), these systems may be vulnerable to a range of cyber attacks. Examples include sybil attacks (an attacker simultaneously forges multiple identities to send CV data) (Alnasser et al., 2019), falsified data attacks (an attacker sends falsified CV data to fool other CV users) (Chen et al., 2018), and denial-of-service attacks (an attacker sends dummy messages to jam the communication channel so other CV users cannot access communication services) (Zeadally et al., 2012). Among all attack types, **falsified data attacks**, are the focus of this dissertation. More specifically, the threat model considered in this dissertation assumes that a hypothetical attacker can send falsified CV data to influence signal control decisions, resulting in a downgraded system performance. Falsified data attacks are chosen for the following three reasons.

First, these types of attacks are practically feasible. By exploiting software vulnerabilities, an attacker could hypothetically hack into his/her own CV's communication device and broadcast falsified CV data. This method is similar to compromising other Electronic Control Units as demonstrated in the literature (Koscher et al., 2010; Checkoway et al., 2011). Alternatively, an attacker could hack into a vehicle's internal network in many ways, such as via the infotainment system (Mazloom et al., 2016). Once the attacker accesses the vehicle's internal network, he/she can take control of numerous vehicle functions (Koscher et al., 2010), including sending falsified CV data. Because the attacker has arbitrary access to his/her own vehicle, falsified data attacks are achievable, similar to spoofing location data on private phones when playing Pokémon Go (Zhao and Chen, 2017).

Second, falsified data attacks are difficult to identify. To ensure trusted communication, the Security Credential Management System (SCMS) (Whyte et al., 2013; Brecht et al., 2018) is being adopted by USDOT. This system requires that the sender signs each CV message with a digital certificate, after which the receiver verifies the signature before using information in the message. An attacker can use a legitimate communication device to transmit falsified CV data with valid digital certificates. In this case, the attacker does not spoof the sender's identity but only modifies the contents of a message (e.g., speed and location data). Falsified CV data can be signed properly to pass an SCMS identity check once received by the RSU, after which these data can be used for signal timing optimization. In other words, the SCMS cannot detect falsified data attacks.

Third, falsified data attacks carry little risk of discovery. An attacker aims to launch effective attacks while seeking to minimize the possibility of being exposed. A falsified data attack does not require the attacker to physically access transportation infrastructure or to connect remotely to the agency's internal network. The attacker only needs to compromise a CV to broadcast falsified messages. Moreover, the compromised CV does not need to stay on the road to launch attacks as long as it is within the RSU's communication range.

Note that this dissertation focuses on evaluating the effects of falsified data attacks on mobility-related issues in CV-TSC systems. Falsified data attacks may only influence the timing (e.g., green time and/or sequence) of traffic signals, whereas phase configurations are not influenced by such attacks. For example, an attacker cannot alter the minimum and maximum green time of each phase, switch the traffic lights into flashing-red status, or cause conflicting phases to be green at the same time. This is because phase configurations are protected by the Malfunction Monitoring Unit (National Electrical Manufacturers Association, 2003) and Cabinet Monitor Unit (AASHTO et al., 2006) in the signal control cabinet.

1.3 Dissertation Framework

Based on the above discussion, this dissertation investigates cyber security problems in CV-TSC systems relative to falsified data attacks. The following three research questions are considered.

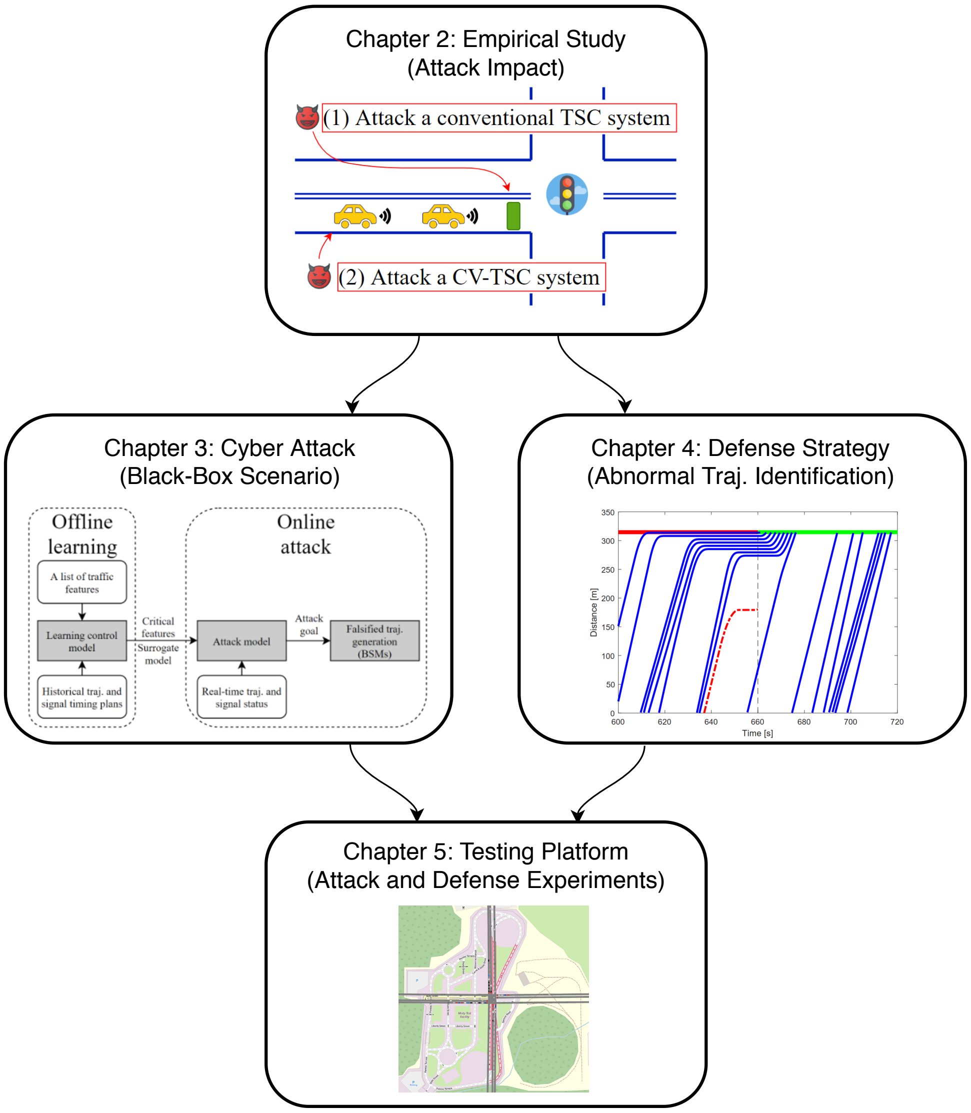
(1) How vulnerable are CV-TSC systems to falsified data attacks?

(2) How falsified data attacks may be perpetrated and how such attacks may influence signal control decisions?

(3) Most importantly, how can engineers mitigate cyber attacks and protect CV-TSC

systems?

By answering these questions, this dissertation attempts to provide a holistic understanding of CV-TSC systems' cyber security problems based on domain knowledge from transportation engineering. As depicted in Figure 1.2, this dissertation consists of the following four studies.



Chapter 2: Empirical Study
(Attack Impact)
(1) Attack a conventional TSC system
(2) Attack a CV-TSC system
Chapter 3: Cyber Attack
(Black-Box Scenario)
Offline learning
A list of traffic features
Learning control model
Historical traj. and signal timing plans
Critical features
Surrogate model
Online attack
Attack model
Attack goal
Falsified traj. generation (BSMs)
Real-time traj. and signal status
Chapter 4: Defense Strategy
(Abnormal Traj. Identification)
Distance [m]
Time [s]
Chapter 5: Testing Platform
(Attack and Defense Experiments)

Figure 1.2: Cyber security framework of CV-TSC systems

Study 1: Empirical Study (Chapter 2)

The objective of this empirical study is to determine whether cyber security concerns in CV-TSC systems are worth investigation. The study assesses the impacts of falsified data attacks on different TSC systems. The classic Cell Transmission Model (Daganzo, 1994, 1995) is applied to model traffic. Two TSC systems are considered, a conventional actuated TSC system and a CV-TSC system. For the former, falsified data attacks are assumed to alter detector actuation; for the latter, falsified data attacks are assumed to change the number of CVs on the road.

Study 2: Cyber Attack (Chapter 3)

The empirical study confirms the vulnerability of CV-TSC systems under falsified data attacks. The second study further investigates how falsified data attacks may be perpetrated. By understanding the attack mechanism, an effective defense strategy can then be developed in the third study. The second study considers a more realistic but challenging black-box attack scenario, which assumes that the control model is unavailable to the attacker. Under this constraint, the attacker has to learn the control model using a surrogate model. With the trained surrogate model, the attacker can then predict signal timing plans based on critical traffic features that are measured from CV data. The attacker can generate falsified CV data (i.e., falsified vehicle trajectories) to alter the values of critical traffic features and thus influence signal control decisions.

Study 3: Defense Strategy (Chapter 4)

Given a thorough understanding of the black-box attack scenario, the third study proposes a defense strategy intended to protect CV-TSC systems by identifying and filtering out falsified vehicle trajectories. The second study reveals that falsified trajectories are generated with a certain attack goal (i.e., altering the values of critical traffic features). These falsified trajectories are behaviorally distinct from normal trajectories. The problem of identifying falsified trajectories is considered an abnormal trajectory identification problem. Inspired by a word embedding method drawn from the Natural Language Processing community, a trajectory embedding model is developed to generate vector representations of trajectory data. The similarity (distance) between each pair of trajectories can be computed based on these vector representations. Hierarchical clustering is applied to identify abnormal (i.e., falsified) trajectories.

Study 4: Testing Platform (Chapter 5)

The fourth study presents a testing platform to evaluate the real-world impact of cyber attacks on CV-TSC systems and the effectiveness of defense strategies. This testing platform is built upon a virtual traffic simulator called VISSIM and real-world transportation infrastructure in Mcity. Attack and defense experiments can be conducted with this testing platform.

1.4 Dissertation Contributions

Collectively, this dissertation contributes to a systematic understanding of cyber security problems facing CV-TSC systems. The empirical study (Chapter 2) evaluates the impacts of falsified data attacks on a conventional TSC system and a CV-TSC system. The cyber attack study (Chapter 3) considers a realistic black-box attack scenario, in which the control model is assumed to be unknown to the attacker. This assumption differentiates this study from the existing literature using a white-box attack scenario. To learn the unknown control model, this dissertation proposes using decision tree models. Attacks can be launched by formulating a simple problem based on learned decision tree models. Moreover, this study formulates a mathematical problem to generate falsified vehicle trajectories. The study of defense strategy (Chapter 4) provides a countermeasure to safeguard CV-TSC systems. This study proposes a data-driven method to identify falsified trajectories. This method only requires trajectory data and does not need external data sources. A trajectory embedding model is developed innovatively to generate vector representations of trajectory data. The proposed method uses vector representations to compute the similarity between trajectories and a hierarchical clustering method to find the abnormal (i.e., falsified) trajectories.

Besides its theoretical contributions, this dissertation also makes practical contributions. This dissertation develops a testing platform (Chapter 5) to perform real-world attack and defense experiments. This testing platform can be used to evaluate the vulnerability of new CV-TSC systems and the effectiveness of new defense strategies before they are deployed in the real world. Moreover, the proposed defense strategy (i.e., abnormal trajectory identification) can be implemented on the infrastructure side as well as on the vehicle side. In general, the defense strategy can be leveraged as a misbehavior detection method for the SCMS and help manage the Certificate Revocation List in the SCMS [1].

To the best of the author's knowledge, this is the first study that comprehensively investigates cyber security problems in TSC systems within a CV environment and holistically integrates attack and defense studies into real-world transportation infrastructure and communication networks.

1.5 Dissertation Organization

The remainder of this dissertation is organized as follows.

Chapter 2 presents an empirical study to measure the consequences of cyber attacks on a conventional actuated TSC system and a CV-TSC system. After a scenario description, the traffic model and attack model are introduced. Numerical examples of cyber attacks on the two TSC systems are then presented.

Chapter 3 investigates how an intelligent attacker can launch falsified data attacks without knowing the control model. The process for learning the control model is introduced, followed by the attack model and the falsified trajectory generation method. A case study is presented that takes I-SIG as the targeted CV-TSC system.

To protect CV-TSC systems, Chapter 4 presents a data-driven method to identify falsified trajectories. The problem formulation is presented. Then the methodology is ex-

[1] The SCMS has the capability of certificate revocation. The SCMS periodically updates and distributes a Certificate Revocation List (CRL). If a legitimate device is identified to be misbehaving or malfunctioning, it would be added to the CRL. Messages sent from a device that is in the CRL would be rejected by other end users.

plained in detail, including the trajectory embedding model, the computation of trajectory similarity and the clustering method. Numerical examples are presented considering different attack goals.

Chapter 5 presents the development of the testing platform. The architecture of the testing platform is described. Attack and defense experiments are then designed and performed based on different attack goals and CV penetration rates.

Finally, Chapter 6 concludes this dissertation with an overview of the work presented along with directions for future research.

CHAPTER 2

Empirical Study: Impact Evaluation of Falsified Data Attacks

2.1 Introduction

As a critical part of transportation infrastructure, existing TSC systems have profound effects on the mobility of urban traffic flow but are highly vulnerable to cyber attacks due to a "systematic lack of security consciousness" (Ghena et al., 2014). As discussed in Chapter 1, for the purpose of this dissertation, a falsified data attack is deemed realistic for three reasons: (1) it is practically feasible; (2) it is difficult to perceive; and (3) it minimizes the chance of attacker identification. Therefore, it is highly plausible that an attacker will launch falsified data attacks against TSC systems. However, it remains unclear whether such attacks can create critical failures in TSC systems.

To address this gap, this chapter presents an empirical study evaluating the adverse effects of falsified data attacks on a conventional TSC system and on a CV-TSC system by measuring the additional travel delay caused by these attacks. The conventional TSC system uses vehicle detector data to perform actuation logic, whereas the CV-TSC system uses a control model adapted from (Sen and Head, 1997; Feng et al., 2015). Falsified data attacks on these two TSC systems are illustrated in Figure 2.1. For the conventional TSC system, this study assumes that an attacker can spoof wireless vehicle detectors and either generate fake vehicle calls or cancel real vehicle calls. For the CV-TSC system, this study assumes that an attacker can add fake or block real communication messages such as BSMs. The number of CVs on the road can thus be manipulated. A traffic flow model is needed to model the transportation network and quantify the effects of cyber attacks. This study uses the Cell Transmission Model (CTM) (Daganzo, 1994, 1995) for two reasons: (1) CTM is a macroscopic traffic model that can be used to efficiently simulate network traffic with thousands of vehicles and attack cases; and (2) compared with other link-based flow and density models, CTM divides the roadway into homogeneous segments so attacks can be launched in different locations (i.e., cells).

The rest of this chapter is organized as follows. Section 2.2 presents the CTM and explains how the two chosen TSC systems are modeled using this traffic model. Section 2.3 explains the attack model, i.e., how falsified data attacks are modeled for these two TSC systems. Section 2.4 presents numerical examples to evaluate the effectiveness of different attack strategies at a hypothetical intersection. Section 2.5 summarizes this chapter.

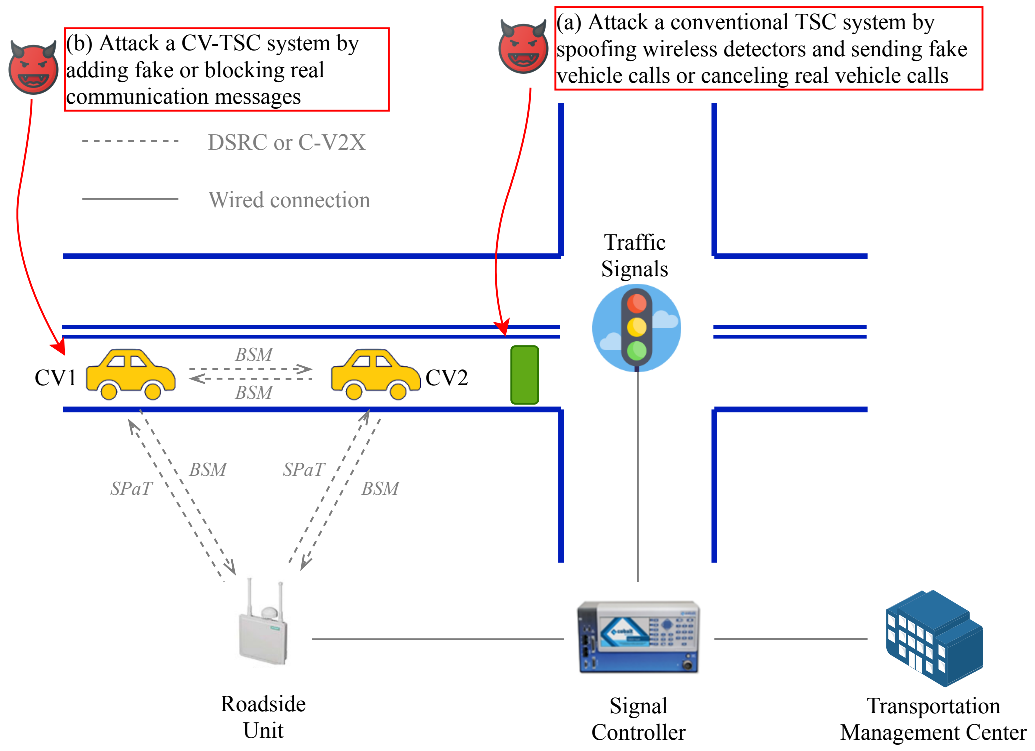

(b) Attack a CV-TSC system by adding fake or blocking real communication messages
(a) Attack a conventional TSC system by spoofing wireless detectors and sending fake vehicle calls or canceling real vehicle calls
DSRC or C-V2X
Wired connection
Traffic Signals
CV1
BSM
BSM
CV2
SPaT
BSM
SPaT
BSM
Roadside Unit
Signal Controller
Transportation Management Center

Figure 2.1: Illustration of falsified data attacks on TSC systems: (a) attack on a conventional TSC system; and (b) attack on a CV-TSC system

2.2 Traffic Model

2.2.1 Cell Transmission Model

CTM is a first-order approximation to Lighthill-Whitham-Richards (LWR) partial differential equation (Lighthill and Whitham, 1955; Richards, 1956). The model assumes a triangular fundamental diagram and discretizes space into homogeneous cells and time into intervals. The cell length is equal to one-time interval multiplied by free-flow speed defined in the fundamental diagram. CTM was originally developed to model highway traffic with a single entrance and exit (Daganzo, 1994). Later the model was extended to represent network traffic (Daganzo, 1995). This allows it to model traffic flows at signalized intersections. A typical intersection in CTM is shown in Figure 2.2.

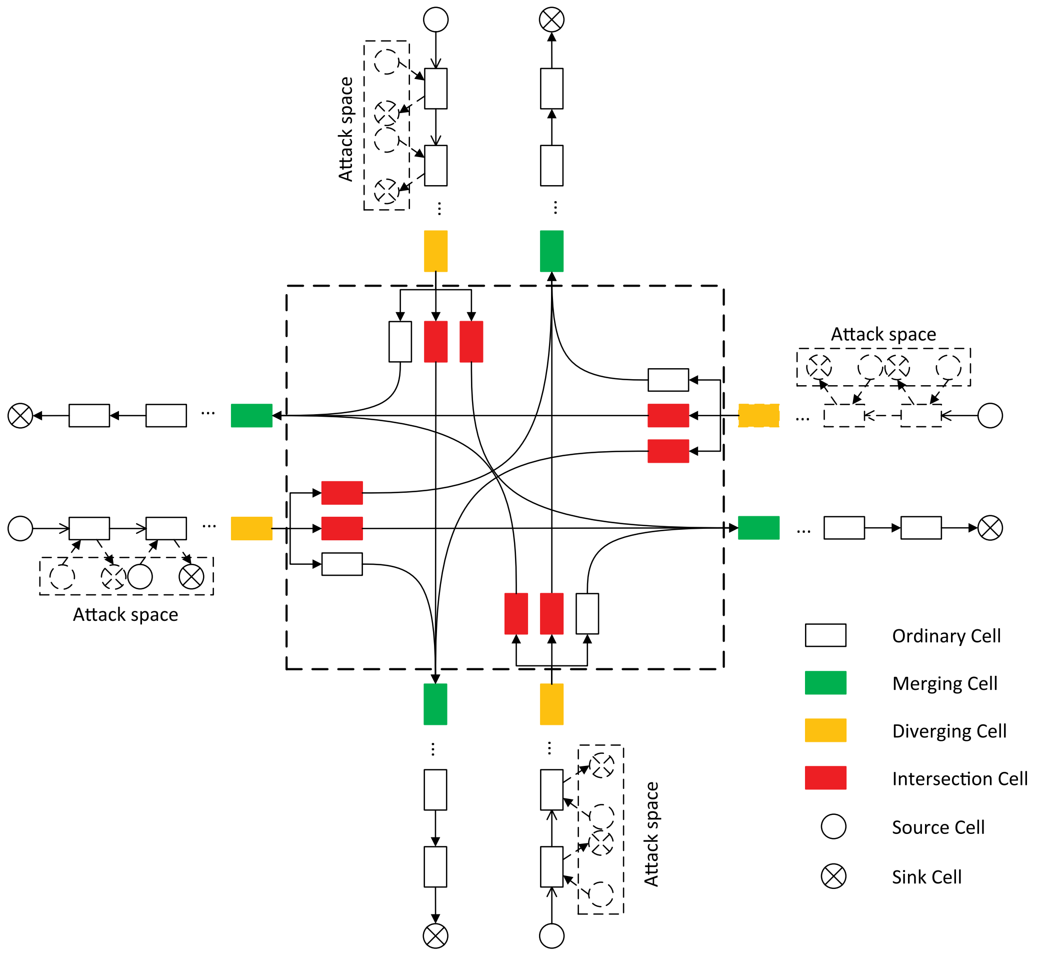


Figure 2.2: Intersection representation of CTM with attack spaces

There are six types of cells: ordinary cell, merging cell, diverging cell, intersection cell, source cell, and sink cell. An ordinary cell has one preceding cell and one following cell and has limited jam density and capacity. A diverging cell has one preceding cell and multiple following cells while a merging cell has multiple preceding cells and one following cell. An intersection cell is similar to an ordinary cell except that the flow is controlled by signal timing. Source cells and sink cells are responsible for generating and exiting vehicles. For the detailed formulation of CTM, refer to Daganzo (1994, 1995).

The parameters of the CTM model in this paper is configured as follows. Free-flow speed v is set to 54 km/h (15 m/s). Backward shockwave speed w is set to 18 km/h (5m/s). The maximum flow rate Q_m is set to 1800 veh/h. The corresponding critical density k_m is 33.33 veh/km. The jam density k_j is 133.33 veh/km. Time step is set to 2 seconds, which is the same to the unit extension time in the conventional TSC system. As a result, the cell length is 30m.

2.2.2 Model the Conventional TSC System with CTM

The conventional TSC system relies on infrastructure-based sensors for vehicle detection. Vehicle detectors are common sensors that are used to detect vehicles and generate service calls to traffic signals. As shown in Figure 2.1, the traffic signal utilizes vehicle detector data to perform actuation logic. To model the conventional TSC system with CTM, this study assumes that stop-bar detectors are installed in intersection cells. The density ratio of cell i at time t can be calculated as $d(i,t) = n(i,t)/N(i)$. Where, $n(i,t)$ is the number vehicles in cell i at time t, and $N(i)$ is the maximum number of vehicles in cell i. A critical density ratio d_c is defined for each intersection cell. The actuation logic is modeled based on d_c. If the density ratio of intersection cell i at time t is less than the critical density ratio, then the current phase is terminated. Otherwise, extend green to next time step.

To determine the best d_c, a series of simulations with different values of d_c were run.

Vehicle delay in cell i at time t is defined as the difference between $n(i,t)$ and the number of vehicles that can be discharged from the cell $y(i,t)$, as show in Equation 2.2.1. This is because in CTM vehicles are either in free-flow speed (discharged to the following cell) or in queuing state (remain in the current cell).

$$D = \sum_{t \in T} \sum_{i \in I} [n(i,t) - y(i,t)] \tag{2.2.1}$$

Both lower and higher critical density ratios result in higher vehicle delay. Lower critical density ratios correspond to longer unit extension times while higher critical density ratios correspond to shorter unit extension times. The lowest delay occurs when $d_c = 0.25$, which is the critical density ratio ($k_m/k_j = 0.25$) that separates free-flow and congestion regime. For vehicle actuation logic, it is appropriate to terminate green when the traffic state changes from congested to free-flow. Therefore, $d_c = 0.25$ is used in case studies.

2.2.3 Model the CV-TSC System with CTM

The control model of the CV-TSC system in this study is adapted from Sen and Head (1997) and Feng et al. (2015). Signal optimization is formulated as a dynamic programming (DP) problem, in which each phase is considered as one stage in DP. A forward recursion is used to calculate performance measures and record the optimal value function. The objective of the forward recursion is to choose an optimal signal timing plan with minimal total vehicle delay. A backward recursion is used to retrieve the optimal solution.

The major difference of the algorithm applied in this study from the original algorithm is the performance function calculation. In previous DP formulations, the performance measures were calculated from an arrival table, which included the estimated time of arrival and requested phase of each vehicle. However, CTM is a macroscopic model in which individual vehicle information is not available. To accurately calculate vehicle delay, a snapshot of the current network condition is taken at the beginning of each signal optimization. A parallel CTM simulation is executed based on the snapshot as the initial network condition to generate vehicle delays in each DP iteration.

The signal optimization algorithm plans as many stages (phases) as necessary until all vehicles in the snapshot pass the intersection cells. A rolling horizon scheme is adopted in which the optimization is performed at the beginning of each phase to include recent vehicle arrivals.

2.3 Attack Model

It is assumed that the attacker has limited resources. For the conventional TSC system, there is a limited number of vehicle detectors that can be spoofed. For the CV-TSC system, there is a limited number of CVs can be manipulated. As a result, the attacker needs to choose a subset of attack space to maximize the utility, which in this study is the total travel delay. Formally, an attack A is defined as:

$$A = (S, \{n'(i,t) \mid \forall i \in S\}) \tag{2.3.1}$$

Where S is the set of cells can potentially be under attack and $n'(i,t)$ is the number of vehicles in the cell under attack. Attacks are conducted through increasing or decreasing the number of vehicles in a cell to mimic the change of stop-bar detector data and CV distribution.

The attack model can be expressed as:

$$\begin{aligned} \max_{A} \quad & D(A) \\ \text{s.t.} \quad & |S| \leq B \\ & n'(i,t) \leq \min(n(i,t) + \epsilon, N(i)) \\ & n'(i,t) \geq \max(n(i,t) - \epsilon, 0) \end{aligned} \tag{2.3.2}$$

The objective function means that the attacker intends to maximize total vehicle delay. The first constraint indicates that the attacker is limited by a budget B. The next two constraints represent the cautiousness of the attacker. The number of vehicles can be changed is limited by a threshold ϵ and road physical limits. If the data deviation exceeds the normal range, the attacker can be detected by defense models easily.

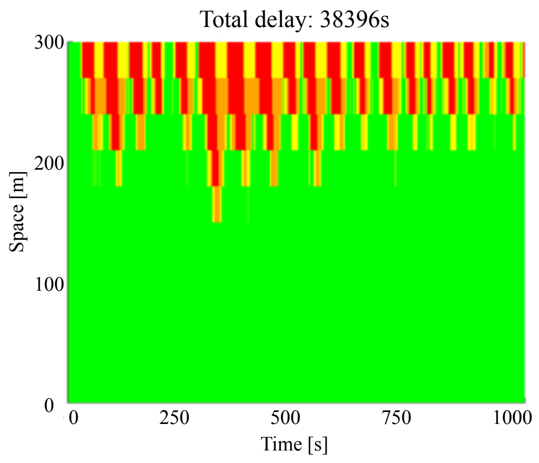
2.4 Numerical Examples

In this section, numerical results based on a hypothetical intersection and insights on the effectiveness of falsified data attacks on the two TSC systems are presented. Both examples are coded in Matlab (MATLAB, 2018).

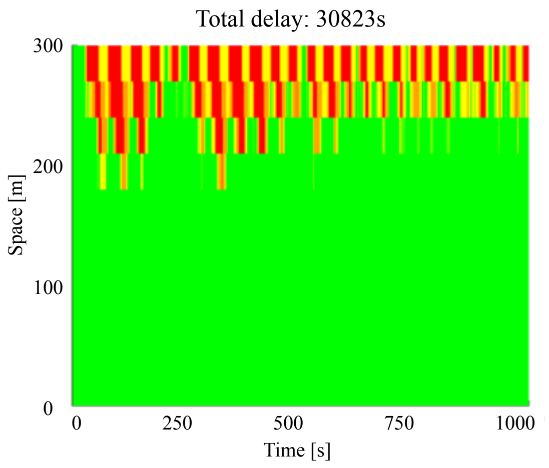
2.4.1 A Hypothetical Intersection

The layout of a hypothetical intersection is shown in Figure 2.2. It is a typical four-leg intersection with all vehicle movements. There are four signal phases: eastbound and westbound left-turn (phase 1), eastbound and westbound through (phase 2), northbound and southbound left-turn (phase 3), and northbound and southbound through (phase 4). Right-turn vehicles are not restricted by traffic signals. The minimum green time is set to 5 time steps (10 seconds) and the maximum green time is set to 20 steps (40 seconds) for each phase. The length of each approach is 10 cells (including the intersection cells), which is similar to the DSRC communication range. Traffic demand is set to 1000 veh/h eastbound/westbound and 800 veh/h northbound/southbound. Vehicle arrivals follow a Poisson distribution. Turning ratios of each approach are the same and set to 0.2/0.7/0.1 for left-turn, through, and right-turn respectively. Simulation runs for 1000 time steps with 100 steps as a warm-up period.

Figure 2.3 shows the total delay and congestion pattern of the eastbound approach for the conventional TSC system and CV-TSC system without attacks. This serves as the baseline for the comparison. Different colors represent different congestion levels, with green to be no congestion and red to be the severest congestion.



(a)



(b)

Figure 2.3: Comparison of traffic density without falsified data attacks: (a) the conventional TSC system; and (b) the CV-TSC system

2.4.2 Attack the Conventional TSC System

For the conventional TSC system, it is assumed that stop-bar detector data can be manipulated by the attacker so that the number of vehicles at intersection cells can be added (generate fake vehicle calls) or subtracted (cancel real vehicle calls). This results in two attack modes $M = 2$. To cause maximum damage, the attacker changes the detector data as much as possible, but within the threshold $\epsilon = 0.5$. Then $n'(i,t)$ is equal to either $\min(n(i,t) + \epsilon, N(i))$ or $\max(n(i,t) - \epsilon, 0)$. The budget B is set to 4 so that all phases can be attacked. To thoroughly analyze the effectiveness of all attack cases, all the possibilities are enumerated. This results in a total number of 80 different cases:

$$\sum_{p=1,2,3,4} C_P^p \times M^p = 80 \tag{2.4.1}$$

Where p is the signal phase index, and P is the total number of phases.

Figure 2.4 shows the total vehicle delay and average total delay by the number of attacking phases. It can be seen from Figure 2.4(a) that the effectiveness of different attacks varies. Figure 2.4(b) shows the trend that attacks cause more vehicle delay when the number of attacking phases increases.

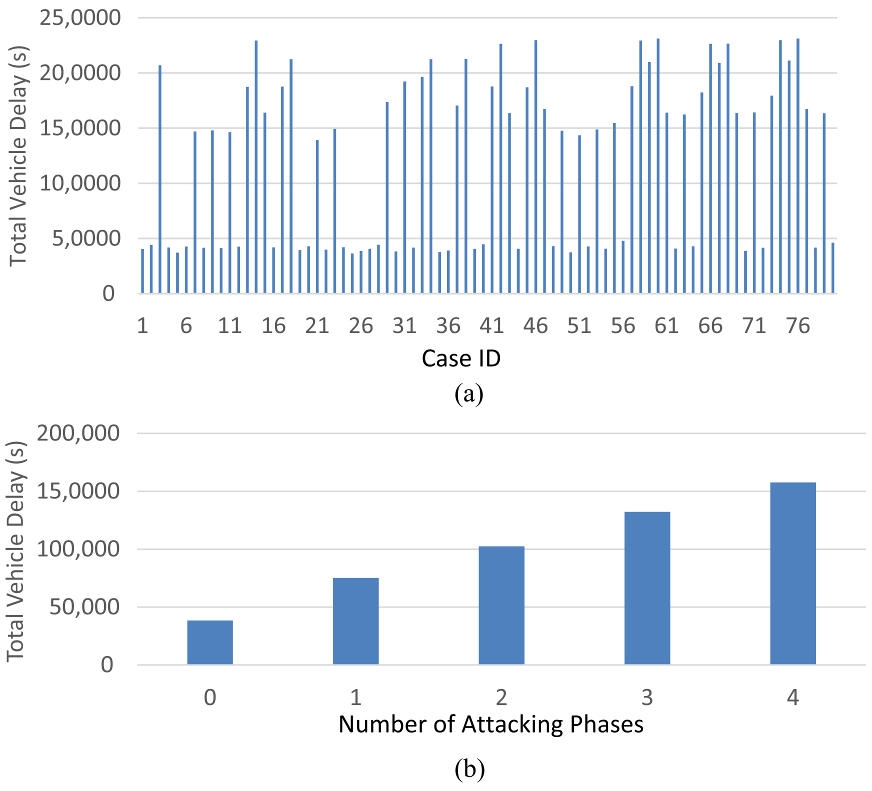
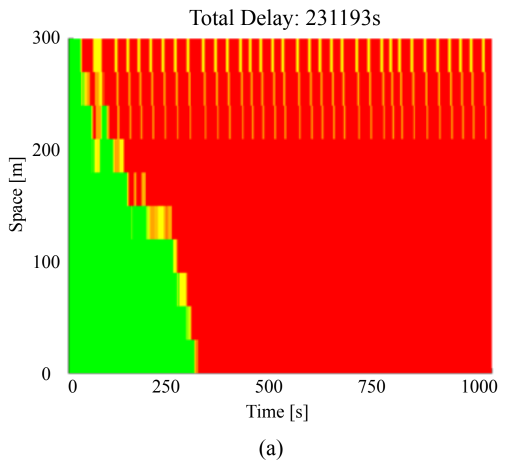


Figure 2.4: Total vehicle delay for the conventional TSC system: (a) by attack cases; and (b) by the number of attacking phases (averaged)

Figure 2.5 shows the comparison between the most effective and least effective attacks at the eastbound approach. The most effective attack occurs when phases 2, 3 and 4 are under attack with subtracting vehicles at intersection cells corresponding to phase 2 and adding vehicles at intersection cells corresponding to phases 3 and 4. The most effective attack results in a six-fold increase of the total delay, compared with the baseline case. The least effective attack occurs when attacking intersection cells related to phases 1 and 3, with subtracting vehicles on both phases. It is interesting to see that the resultant total delay (36441s) is even smaller than the baseline case (38396s). This indicates that the attack improves the system performance. The reason is that actuation control logic is not the optimal control strategy. In certain cases when a phase is green with lower demand while other phases are red with higher demand, it is more efficient to terminate the lower demand phase earlier to serve other phases. In this case, phases 1 and 3 are left-turn phases with lower demand. Subtracting vehicles shortens both phases, therefore more green time is given to higher demand phases 2 and 4.

2.4.3 Attack the CV-TSC System

For the CV-TSC system, the control algorithm utilizes CV data to generate optimal signal timing plans. Every cell within the communication range can be potential targets. It is assumed that the attacker is only interested in manipulating the number of vehicles in ingress cells because vehicles in egress cells do not affect the signal optimization. The attacker can add or subtract vehicles at different number of approaches with the maximum number of attacking approaches $B = 4$. If the attacker decides to attack one approach,



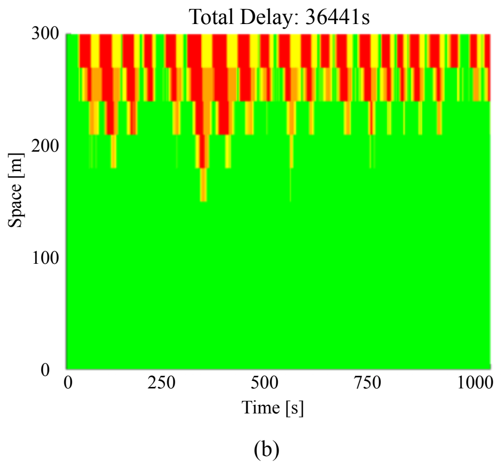
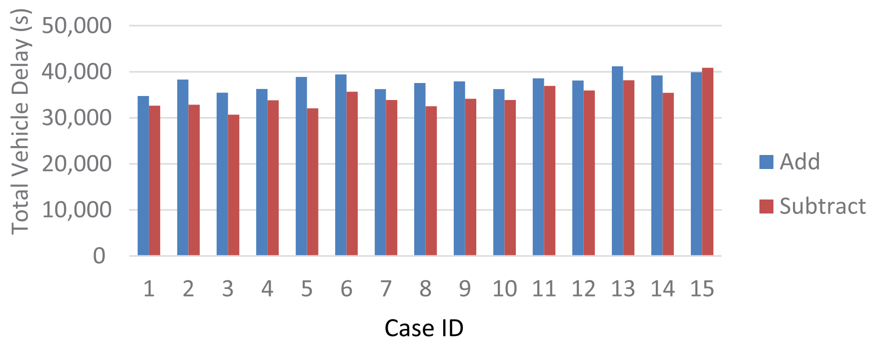


Figure 2.5: Comparison of traffic density with falsified data attacks (conventional TSC system): (a) the most effective attack (Case 60); and (b) the least effective attack (Case 25)

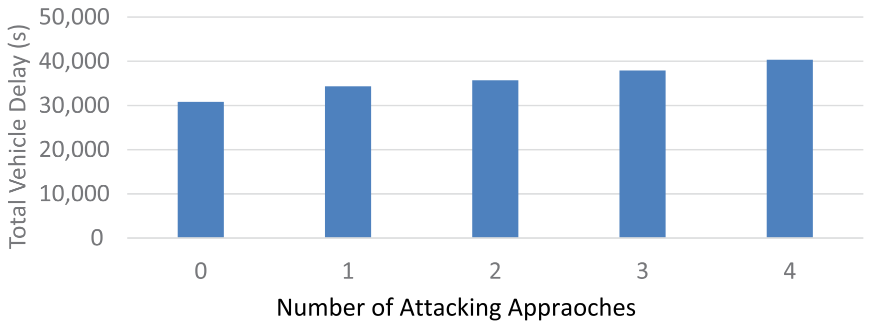
then all ingress cells on that approach are affected. Because of limited resources, it is assumed that the attacker can only add fake or block real communication messages, but cannot do both at the same time. As a result, totally 30 attack cases are generated. The threshold ϵ is also set to 0.5.

Figure 2.6 shows the total vehicle delay of all attack cases and average total delay by the number of attacking approaches. Figure 2.6(a) compares total delay of adding vehicles or subtracting vehicles when attacking the same approach(s). In general, adding falsified vehicles is more effective than removing real vehicles. Because adding vehicles would increase delay for all other phases, while subtracting vehicles only increase delay for the current phase. Figure 2.6(b) shows a similar pattern that the average total delay increases with the number of attacking approaches.

Another finding is that attacking the CV-TSC system is far less effective than at-



(a)



(b)

Figure 2.6: Total vehicle delay for the CV-TSC system: (a) by attack cases; and (b) by the number of attacking approaches (averaged)

tacking the conventional TSC system with the same attack intensity ϵ. For the CV-TSC system, the most effective attack causes 41199s vehicle delay, which is 33.66% more than the baseline case. For the conventional TSC system, however, the most effective attack increases the total delay by a factor of six. For the CV-TSC system, all ingress cells (including intersection cells) are affected. While for the conventional TSC system, only intersection cells are affected. However, results suggest that the CV-TSC system is more robust than the conventional TSC system. The CV-TSC system tries to minimize total delay under the impact of attacks based on the inputs from all ingress cells, while the conventional TSC system only accommodates instantaneous arriving flow at intersection cells, but does not have an overall picture of current traffic conditions.

2.5 Chapter Summary

This chapter presents an empirical study investigating the effects of falsified data attacks on two TSC systems. The CTM was used to model traffic. For a conventional TSC system with actuated control logic, falsified data attacks were assumed to either generate fake vehicle calls or cancel real vehicle calls. For a CV-TSC system, falsified data attacks were assumed to change the number of CVs on the road. The attacker's primary objective was to maximize system delay under constraints related to budget and attack intensity. The most effective attack on a conventional TSC system was found to increase the total delay by a factor of six, while the most effective attack on a CV-TSC system led to a 33.66% greater delay over baseline.

This empirical study confirms that CV-TSC systems are vulnerable to falsified data attacks. However, the attack model presented in this chapter is fairly naïve; the attacker can enumerate all possible attack cases to identify an optimal attack strategy. From an attacker's point of view, attack execution takes a long time and is highly inefficient. This attack model is thus ill-suited to real-time attacks. A more sophisticated and realistic attack model will be introduced in the next chapter.

CHAPTER 3

Cyber Attack: A Black-Box Attack Scenario

3.1 Introduction

In the previous chapter, a falsified data attack was designed to manipulate the input data (i.e., BSMs) of a CV-TSC system. Results indicated that this CV-TSC system was vulnerable to such attacks. However, the attack model proposed in Chapter 2 is inefficient and cannot be used to launch real-time attacks because the optimal attack strategy is identified by enumerating all possible attack cases. Further, most other work concerning the cyber security problem has considered a "white-box" attack scenario (Laszka et al., 2016; Reilly et al., 2016; Ernst and Michaels, 2017; Chen et al., 2018; Yen et al., 2018; Ganin et al., 2019; Perrine et al., 2019), in which the attacker is assumed to have full access to the TSC system and/or the control model. For instance, Chen et al. (2018) also focused on I-SIG attacks. In their study, the attacker had access to the source code of I-SIG and could identify flaws in the control logic. This is a special case because the I-SIG code is open-source. However, for most commercial TSC systems deployed in the real world, source codes are unavailable to attackers. The attack model presented in Chapter 2 also involved a white-box attack scenario, as running the CTM simulation requires knowing the control model. The major drawback of this white-box attack scenario is that it cannot realistically evaluate the effects of cyber attacks. An attacker having full knowledge of the control model (i.e., the source code) is a strong and unrealistic assumption, hence the need to investigate cyber threats in real-world scenarios where the control hardware and control model are inaccessible.

This chapter explores a more realistic but challenging black-box attack scenario and investigates how falsified data attacks may be executed in real time. The term "black-box" means that the attacker neither knows the details of the TSC system nor has physical access to the system. This "black-box" attack includes two steps. The first step is an offline learning process, in which the attacker identifies critical traffic features and learns the signal control model using a surrogate model. Critical traffic features are selected from a list of traffic features, such as queue length, number of approaching vehicles, and travel time, which are measured using trajectory data. In TSC systems, these data function as input for signal optimization. The surrogate model predicts the signal timing plan based on critical traffic features. In a CV environment, all vehicles and infrastructure broadcast communication messages. The attacker can receive the same messages as all other vehicles and infrastructure receive; therefore, the surrogate model can be trained using observed trajectory data (i.e., BSMs) and resultant signal timing plans (i.e., SPaT).

In the second step, the attacker launches real-time falsified data attacks based on the critical traffic features and surrogate model obtained in the first step. It is assumed that the attacker launches the attack in the vicinity of an intersection to be able to receive all trajectories from nearby CVs in real time. With the received trajectories, the attacker can predict the signal timing plan of the CV-TSC system via the surrogate model. The predicted signal timing plan is called "pseudo-optimal" because it is not the exact original timing plan generated by the CV-TSC system. The attacker's objective is to increase the system delay. To do so, the attacker inserts falsified trajectories to modify the values of critical traffic features. For example, the attacker can insert a falsified stopped vehicle trajectory and increase the queue length. The attacker also formulates a mathematical problem (i.e., attack model) with the objective to distort the pseudo-optimal timing plan to the greatest extent. The optimal solution to this problem returns the altered values of critical traffic features (i.e., the attack goal). Finally, the attack goal is embedded into another mathematical problem to generate falsified trajectories, in the form of BSMs, and is then broadcast through the OBU. The overall process is illustrated in Figure 3.1 (in this case, the attacker is assumed to have compromised CV2).

The rest of this chapter is organized as follows. Section 3.2 introduces the off-line learning process. Section 3.3 explains the attack model, and Section 3.4 describes how the attacker formulates a mathematical problem to generate falsified CV trajectories. Section 3.5 presents a comprehensive case study evaluating the effects of black-box attacks on I-SIG. Section 3.6 summarizes this chapter.

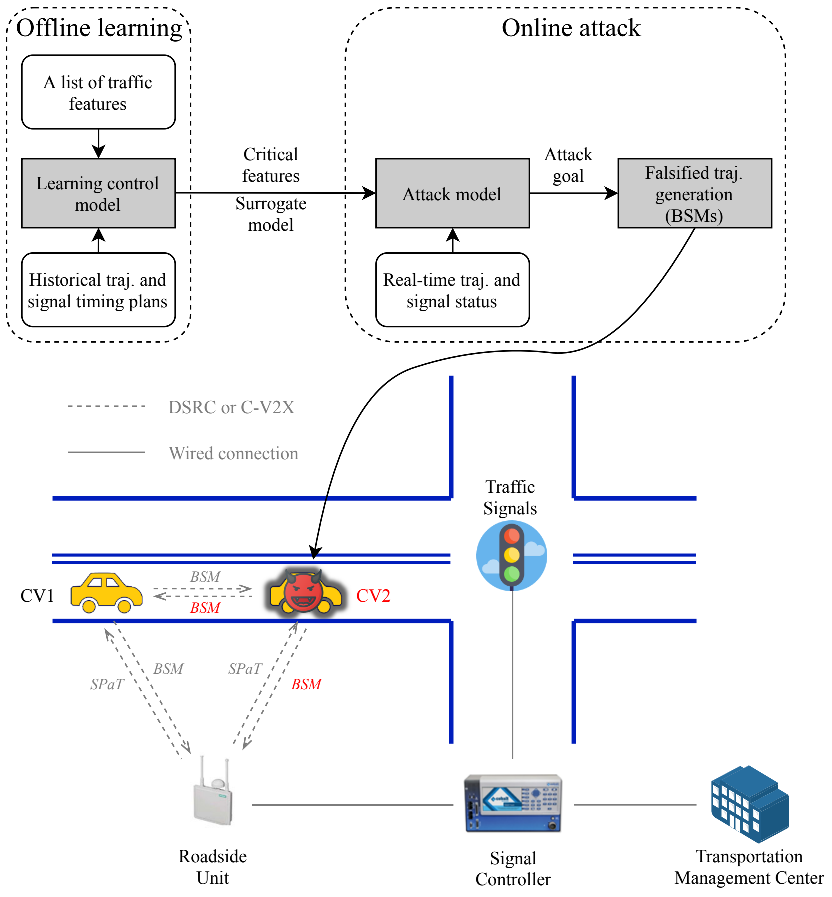


Figure 3.1: Illustration of the black-box attack

3.2 Learning Control Model

In a real-world implementation, the CV-TSC system utilizes vehicle trajectories (obtained from BSMs) as the input and generates optimal signal control decisions. Because all the communications in the vehicular network are in broadcast mode, both vehicle trajectories and signal control decisions are observable to the attacker in the forms of BSMs and SPaT messages respectively. The actual signal control model, however, is unknown to the attacker. The attacker can adopt a surrogate model to learn the signal control model.

The surrogate model takes the same trajectories as the input and outputs predicted signal timing plan. Historical BSM and SPaT data can be used to train the surrogate model. The attacker uses the surrogate model as the replacement of the real control model when launching attacks. The whole learning process is illustrated in Figure 3.2.

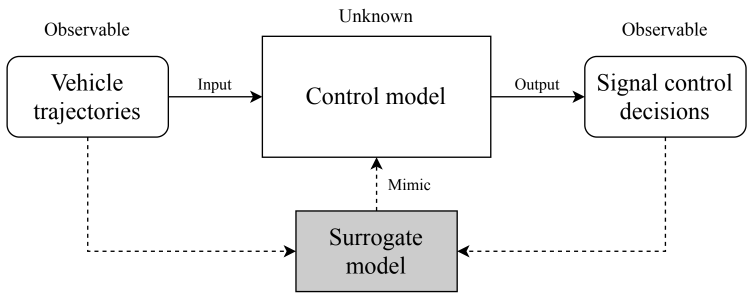


Figure 3.2: The process of learning control model

3.2.1 Traffic Signal Setting

This study assumes that the CV-TSC system uses a ring-barrier phasing. The ring-barrier structure (Koonce and Rodegerdts, 2008) illustrated in Figure 3.3 is the standard traffic signal setting in North America. Starting from the major street and moving clockwise, the through phases are labeled as phases 2, 4, 6, and 8. Starting from the left-turn phase that is next to phase 6 and moving clockwise, the left-turn phases are labeled as phases 1, 3, 5, and 7. Ring 1 includes phases 1 to 4 and ring 2 includes phases 5 to 8. A barrier separates major street phases (phases 1, 2, 5, and 6) from minor street phases (phases 3, 4, 7, and 8). A barrier may also refer to the four phases of the major street or the four phases of the minor street. The phase that operates first within a barrier is called lead phase and the other one is called lag phase, therefore a barrier includes two lead phases and two lag phases. Usually the signal optimization algorithms change phase sequence and allocate green time of each phase to minimize/maximize predefined performance indexes.

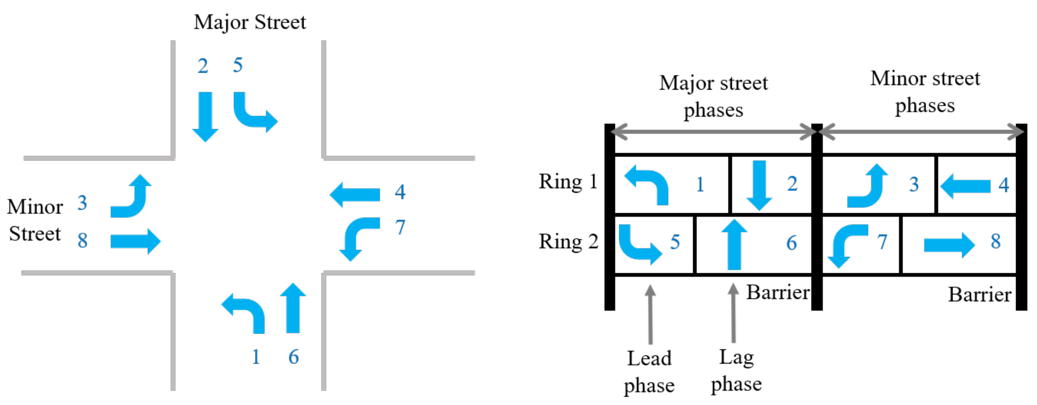


Figure 3.3: Illustration of the ring-barrier structure

3.2.2 Surrogate Model

A signal timing plan includes two parts, green time of each phase and phase sequence. The prediction of green time can be considered as a regression problem because green time is continuous. In contrast, the prediction of phase sequence is a classification problem since there is a finite number of possibilities for phase sequences. In this study, decision tree regression/classification (Breiman, 2017) is adopted to be the surrogate model that could potentially be leveraged by the attacker. Decision tree models are chosen because they are easy to implement and their output always falls within the feasible ranges, i.e., minimum and maximum green time. Most importantly, decision tree models possess inherent "if-then-else" structures and can effectively map nonlinear relationships, making signal control algorithms particularly easy to fit into programmatic structures.

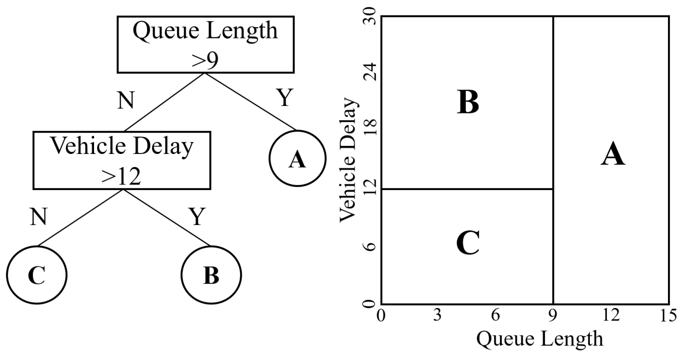


Figure 3.4: Illustration of the decision tree model

The decision tree model is briefly introduced. The goal of a decision tree regression model is to predict y, the green time of a phase or the length of a barrier, based on a d-dimensional feature vector $\boldsymbol{X} = [x^1, x^2, ..., x^d]^\mathsf{T}$, which are extracted from trajectories (see Section 3.2.3 for more details on features). The training data consists of n observations with their corresponding labels $\{y_1, y_2, ..., y_n\}$ and features $\{\boldsymbol{X_1}, \boldsymbol{X_2}, ..., \boldsymbol{X_n}\}$. During the learning process, the decision tree algorithm partitions the entire feature space into different sub-regions based on the training dataset. Figure 3.4 shows a simple example of a trained decision tree with two features. In this example, the first step is to divide the entire space into two sub-regions according to whether queue length is greater than 9. Similarly, in the second step, the left region is further divided into two sub-regions according to whether vehicle delay is greater than 12. Every step is called branching. For each sub-region, mean squared error (MSE), presented in Equation 3.2.1, is calculated to evaluate the prediction performance of that branched sub-region based on the training data:

$$e_D = \sum_{j \in D} \frac{1}{|D|}(y_j - y_D)^2 \tag{3.2.1}$$

where D is the set of all observations in this sub-region; e_D is the MSE of set D; $|D|$ is the cardinality of set D; and y_D is the mean value of the labels in set D.

Each time one performs branching, a node is split into two nodes. Each node represents a sub-region that contains data satisfying associated conditions. For each branching, the parent node is denoted as D_p and the two child nodes are D_{C1} and D_{C2}. Then the reduction in MSE due to a branching from a parent node into two child nodes, ΔI, is defined as follows in Equation 3.2.2:

$$\Delta I = e_{D_p} - e_{D_{C1}} - e_{D_{C2}} \tag{3.2.2}$$

All the observations in the parent data set are used as splitting candidates for the next level of branching. One can enumerate all the possible features and possible splits and calculate the corresponding ΔI. The feature and the split with the maximum ΔI are chosen to branch the parent node. The branching process is repeated until the maximum number of iteration is reached.

For predictions of green time, one can start from the top of the tree, which is the root node, and find the path down to the final node, called the leaf node, based on the criteria of each node. The mean value of the labels in the final data set is the predicted green time.

The process for the prediction of phase sequence is similar to that of green time.

Because it is a classification problem, the majority vote of the labels is taken as the prediction.

3.2.3 Critical Traffic Features

Different CV-TSC systems use different objectives and performance indexes to optimize the signal timing plan. Since the objectives are typically functions of one or more traffic features, the signal timing plan should be closely related to these associated traffic features, e.g., queue length (a signal controller allocates green time based on the queue length of each phase) and headway (a signal controller terminates a green phase when there is a large headway). A list of common traffic features applied in existing studies include queue length (QL) (Priemer and Friedrich, 2009), number of approaching vehicles (NAV) (Goodall et al., 2013), headway (HW) (Koonce and Rodegerdts, 2008), estimated time of arrival (ETA) (He et al., 2012, 2014), vehicle delay (VD) (Wu et al., 2017), and flow rate (FR) (Zheng et al., 2018).

For a particular CV-TSC system, not all traffic features are utilized to optimize the signal timing. The traffic features that determine the signal timing plan are defined as critical features. When falsified data alter the values of these critical features, signal control decisions are changed accordingly. As a result, the attacker needs to identify the critical features that have a significant impact on the signal timing plan before launching attacks. Identifying critical features from the list of features is a feature selection problem. In this study, a sequential forward selection algorithm (SFS) is applied (Bow, 2002).

Starting from an empty feature set, SFS greedily searches for the best features that can improve the prediction performance. John et al. (1994)'s study suggests using SFS for identifying useful features and shows that SFS can improve the performance of decision tree models. The pseudo code of SFS is illustrated in Algorithm 1. The output of SFS is a set that contains all the critical features. Note that in Section 3.2.2, only critical features are used as the input features of the decision tree models.

Algorithm 1: Sequential forward selection algorithm

```
Initialize two sets, $S$ and $R$. $S$ contains a complete list of features and $R$ is empty.
  Denote $e(Q)$ as the error when using feature subset $Q$. Initialize $e(R)$ to be a
  large number;
while $S$ is not empty do
    find $s^* = \min_{s \in S} e(s \cup R)$;
    if $e(s^* \cup R) < e(R)$ then
        Remove $s^*$ from $S$;
        Add $s^*$ to $R$;
    else
        break while;
    end
end
Result: Set $R$ contains all the critical features
```

3.3 Attack Model

In the previous section, the attacker has obtained the surrogate model $\boldsymbol{f}(\cdot)$. Now the attacker can launch cyber attacks by broadcasting falsified trajectories using the compromised communication device (i.e., OBU). Based on the received trajectories, the attacker can evaluate the observed critical traffic features $\boldsymbol{X_o}$ (e.g., queue length) and use the surrogate model to predict the signal timing plan, i.e., $\boldsymbol{f}(\boldsymbol{X_o})$. $\boldsymbol{f}(\boldsymbol{X_o})$ is referred to as the pseudo-optimal timing plan because it is not the exact timing plan generated from the actual control model, but a plan predicted by the trained surrogate model. By injecting falsified trajectories, the attacker tries to alter the values of the critical features from $\boldsymbol{X_o}$ to $\boldsymbol{X_a}$. Similarly, the attacker can predict the signal timing plan with the altered critical feature, i.e., $\boldsymbol{f}(\boldsymbol{X_a})$. The dissimilarity between the pseudo-optimal timing plan and the timing plan under attack can be computed using the L2 norm. The attacker's objective is to maximize the dissimilarity by generating falsified trajectories that can alter the values of the critical features, as shown in the following problem (**P1**):

P1:

$$\max_{\boldsymbol{X_a}} \quad \|\boldsymbol{f}(\boldsymbol{X_o}) - \boldsymbol{f}(\boldsymbol{X_a})\|_2 \tag{3.3.1}$$

$$\text{s.t.} \quad \boldsymbol{X_a} \in \Omega_{\boldsymbol{X_a}|\boldsymbol{X_o}} \tag{3.3.2}$$

In **P1**, the feasible region $\Omega_{\boldsymbol{X_a}|\boldsymbol{X_o}}$ is dependent on $\boldsymbol{X_o}$. For example, after injecting a falsified stopped vehicle (a falsified vehicle has a legitimate trajectory in the form of BSMs but is not physically on road), the new queue length cannot be smaller than the originally observed queue length.

The falsified trajectories are mixed with the regular trajectories. The RSU collects all the trajectories and uses them as input data for real-time traffic signal optimization. The generated signal timing plans are influenced by the falsified trajectories, and thus are no longer optimal. Vehicles spend extra time passing the intersection and hence the total travel time is increased.

This attack model aligns with a recent study on black-box attacks against unknown machine learning models (Papernot et al., 2017). By using a surrogate model, the attacker in that study crafts adversarial images to fool a target model so that the target model would output erroneous predictions.

3.4 Falsified Trajectory Generation

In order to launch a falsified data attack, the attacker has to generate a valid trajectory (i.e., continuously broadcast BSMs). The location, speed, and acceleration rate in the falsified trajectory should be consistent and satisfy kinematic constraints (Wong et al., 2019). Otherwise, a defender can easily identify the falsified trajectory by checking the consistency between location, speed, and acceleration rate. Also, the falsified trajectory should avoid conflicting with other trajectories (i.e., the falsified vehicle should not be too near to a real CV). Otherwise the falsified trajectory would be easily identified by the misbehavior detection in the SCMS (Crash Avoidance Metrics Partners (CAMP) LLC). What's more, the falsified trajectory should achieve a certain attack goal. For example, the attacker in Chen et al. (2018) launches falsified data attacks on I-SIG by changing the speed and location data in the trajectory and therefore manipulating arrival time or queue length. The attack goal comes from the optimal solution to **P1** (i.e., to alter the the values of the critical features). Considering all these, this dissertation envisions that the attacker can solve the following problem (**P2**) to generate a falsified trajectory. The notations are summarized in Table 3.1.

P2:

$$\min_{d(t),v(t),a(t)} \quad \sum_{t=t_\mathrm{s}}^{t_\mathrm{e}} (d_\mathrm{l}(t-\tau^w) - d^w - d(t))^2 \tag{3.4.1}$$

$$\text{s.t.} \quad v(t) \times \Delta t = d(t) - d(t-\Delta t) \qquad \forall t \tag{3.4.2}$$

$$a(t) \times \Delta t = v(t) - v(t-\Delta t) \qquad \forall t \tag{3.4.3}$$

$$d_\mathrm{f}(t) + d_\mathrm{o} \leq d(t) \leq d_\mathrm{l}(t) - d_\mathrm{o} \qquad \forall t \tag{3.4.4}$$

$$0 \leq v(t) \leq v_\mathrm{f} \qquad \forall t \tag{3.4.5}$$

$$a_\mathrm{min} \leq a(t) \leq a_\mathrm{max} \qquad \forall t \tag{3.4.6}$$

$$v(t_\mathrm{s}) = v_\mathrm{s} \tag{3.4.7}$$

$$d(t_\mathrm{s}) = d_\mathrm{s} \tag{3.4.8}$$

$$g(d(t), v(t), a(t)) = \boldsymbol{X}^* \tag{3.4.9}$$

Table 3.1: Notations for generating falsified trajectories

$d(t)$	Location of the falsified vehicle at time t
$v(t)$	Speed of the falsified vehicle at time t
$a(t)$	Acceleration rate of the falsified vehicle at time t
t_{s}	Start time of the falsified trajectory
t_{e}	End time of the falsified trajectory
τ^w	Time displacement in Newell's car following model
d^w	Space displacement in Newell's car following model
Δt	Time interval
$d_{\text{l}}(t)$	Location of the leading vehicle (the vehicle in front of the falsified vehicle) at time t
$d_{\text{f}}(t)$	Location of the following vehicle (the vehicle behind the falsified vehicle) at time t
d_{o}	Safety distance
v_{f}	Free-flow travel speed
a_{min}	Minimum acceleration rate of the falsified trajectory
a_{max}	Maximum acceleration rate of the falsified trajectory
v_{s}	Initial speed of the falsified trajectory
d_{s}	Initial position of the falsified trajectory
$g(\cdot)$	Mathematical representation of the attack goal
$\boldsymbol{X}^*$	The optimal solution to **P1**

Decision variables $d(t), v(t), a(t)$ represent the location, speed, and acceleration rate of the falsified vehicle at time t. The objective function 3.4.1 is based on Newell's car following model (Newell, 2002). $d_{\mathrm{l}}(t)$ denotes the location of the leading vehicle at time t. $d_{\mathrm{l}}(t-\tau^{w})-d^{w}$ denotes the location of the leading vehicle with a time displacement τ^{w} and a space displacement d^{w}. The objective function essentially means that the falsified vehicle chases the leading vehicle based on Newell's car following model. This objective function ensures the falsified trajectory to be like a real trajectory. Constraint 3.4.2 and 3.4.3 describe vehicle dynamics, where Δt is the time interval. Constraint 3.4.4, 3.4.5 and 3.4.6 are the bounds for vehicle location, speed, and acceleration rate. In particular, constraint 3.4.4 guarantees that the falsified trajectory $d(t)$ keeps a safety distance d_{o} from leading trajectory $d_{\mathrm{f}}(t)$ and following trajectory $d_{\mathrm{l}}(t)$ at any time t. This constraint prevents the falsified trajectory from conflicting with other trajectories. As mentioned in Chapter 1, the SCMS monitors the behavior of each end user. If conflicting trajectories are detected, the SCMS would revoke the certificate of the device. Therefore conflicting trajectories would be automatically considered as invalid trajectories. Constraint 3.4.7 and 3.4.8 set the initial values for the falsified trajectory. The falsified vehicle should enter the intersection area from a certain distance (e.g., communication range) with a certain speed (e.g., free-flow speed). Constraint 3.4.2 to 3.4.8 ensures that the falsified trajectory is a valid trajectory. Finally, constraint 3.4.9 represents the general form of the attack goal. If there is no attack goal, the attacker simply replays the trajectory of the leading vehicle with a time delay.

3.5 Case Study

The section presents a case study in simulation. This case study considers the black-box attack scenario and launches falsified data attacks on a CV-TSC system. I-SIG system from the MMITSS project is selected as the targeted CV-TSCS system (USDOT, 2019b). Both simulation and field experiments have demonstrated the effectiveness of I-SIG in terms of delay reduction and mobility improvement (Feng et al., 2015). Note that this case study only evaluates the attack model (i.e., the case study assumes that the attacker is able to change the values of the critical traffic features). The falsified trajectory generation method will be evaluated in Chapter 5.

3.5.1 Control Model of I-SIG

The control model of I-SIG system is briefly introduced in this subsection. At the beginning of each barrier, I-SIG takes a snapshot of the trajectories received from all the CVs within the RSU's communication range. Each trajectory is converted to ETA, which is calculated as the CV's distance to stop bar divided by its speed. Based on the ETAs, I-SIG solves a two-level optimization problem to find the optimal signal timing plan. At the lower level, I-SIG solves a utility minimization problem given the barrier length. The outputs of the lower level are the optimal green time and phase sequence. At the upper level, a dynamic programming (DP) problem is formulated with the objective to minimize total vehicle delay or total queue length. The decision variable at the upper level is the barrier length, which is considered as a stage in the DP formulation. Ideally, I-SIG should plan as many stages as needed so that all the vehicles can be properly served. For real-world implementations, however, I-SIG plans only two stages (i.e., one signal cycle) because of computational limitations in the RSU and real-time performance requirement. I-SIG then executes the timing plan of the first stage (the four phases in the current barrier) and arranges the phase sequence of the second stage (the four phases in the next barrier). When a new barrier starts, I-SIG repeats this optimization process. For more details about the control model, please refer to Feng et al. (2015) and Sen and Head (1997).

3.5.2 Simulation Setup

A simulation environment is built using Matlab (MATLAB, 2018). A typical 4-leg intersection with eight phases is modeled. Each approach has one left-turn lane and one through lane. For simplicity, right-turn lanes are not explicitly modeled. A snapshot of the simulation environment is displayed in Figure 3.5. The crosses represent left-turn vehicles and circles represent straight-through vehicles. Vehicles are generated at the edges of the figure and move towards the center (i.e., the intersection). The CV penetration rate is 100%. The car-following model from the NGSIM project is used to model vehicle motions (Yeo et al., 2008). The minimum green time and the maximum green time are set to be 5 seconds and 30 seconds for each phase, respectively. The transition time between phases (i.e., the yellow and red clearance time) is 4 seconds. The traffic demand for each movement is 400 vehicles per hour. The communication range is set to be 300 meters. The free-flow speed is 15.64 meters per second (i.e., equivalent to 35 miles per hour).

The resolution of the simulation is 10 Hz, which is consistent with the frequency of CV communication (SAE International, 2016). Falsified data attacks are launched every time I-SIG optimizes the signal timing plan.

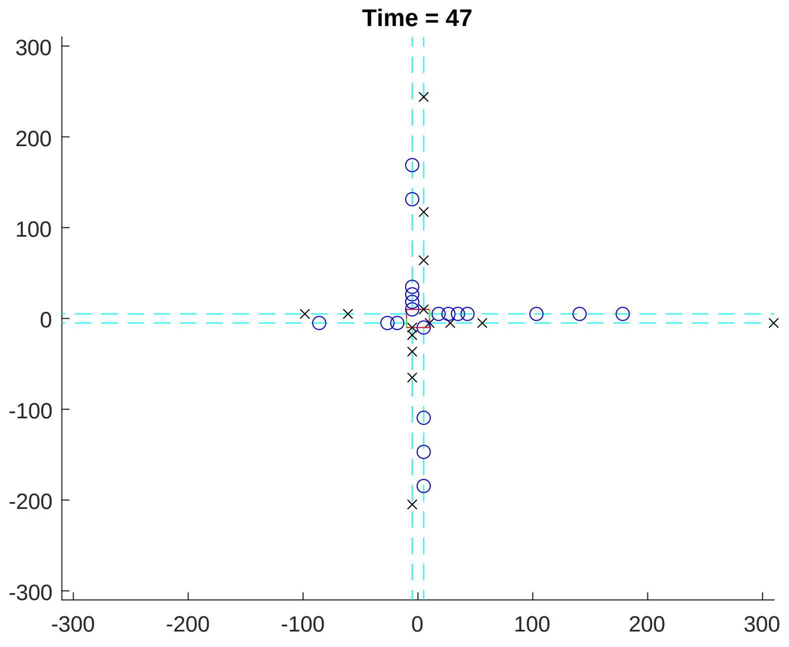


Figure 3.5: A snapshot of the simulation environment

3.5.3 Learning Control Model

Because I-SIG only executes the optimized signal timing for one barrier each time, the surrogate model only needs to predict the green time of the four phases in the current barrier. Phase sequence can be obtained directly from SPaT messages.

The surrogate model consists of two decision tree regression models. The output of the first decision tree model (labeled as Tree 1) is the barrier length (i.e., lead phase plus lag phase). The second decision tree model (labeled as Tree 2) outputs the green time of a lead phase. The green time of a lag phase is calculated by subtracting that of the corresponding lead phase from the barrier. The detailed definitions of the potential traffic features for the two decision tree models are shown in Table 3.2.

Table 3.2: Definitions of the traffic features associated with trees 1 and tree 2

Traffic Feature	Tree 1 (Barrier)	Tree 2 (Lead Phase)
Queue length (QL)	The maximum QL of Ring 1 and Ring 2 (lead phase plus lag phase, the same thereafter)	QL of the lead phase
Number of approaching vehicles (NAV)	The maximum NAV of Ring 1 and Ring 2	NAV of the lead phase
Headway (HW)	Time of arrival of the first large HW (2 seconds) of the two lag phases	Time of arrival of the first large HW (2 seconds) of the lead phase
Estimated time of arrival (ETA)	ETA of the last vehicle of the two lag phases	ETA of the last vehicle of the lead phase
Vehicle delay (VD)	Average VD of the two lag phases	Average VD of the lead phase
Flow rate (FR)	The maximum FR of Ring 1 and Ring 2	FR of the lead phase

A 30-hour simulation is run to generate a data set needed for both training and validation. Totally, 2206 optimizations are conducted. Mean absolute error (MAE), mean absolute percentage error (MAPE) and root mean square error (RMSE) are utilized to quantify errors for a given set of traffic features. Monte Carlo cross validation (Xu and Liang, 2001) is applied. 80% of the data are randomly chosen for training, while the remaining 20% are used for validation. The 80-20 process is repeated 10 times and the mean errors are recorded. The SFS is applied and the results are shown in Table 3.3. In the first round, only one feature is used for fitting the decision tree models. The model with NAV has the least error for both trees. Therefore, NAV is added to the critical feature set. In the second round, two features are used for fitting the decision tree models, with one feature fixed to be NAV. The model with NAV and ETA has the least error. Thus, ETA is chosen as the second feature and added to the critical feature set. This process is repeated to find the third feature. However, the models with three features are all worse than the best model in the second round. Thus, SFS stops searching. NAV and ETA are identified as critical features.

Table 3.3: Applying SFS for identifying critical traffic features

	Tree 1 (Barrier)				Tree 2 (Lead Phase)			
	Feature set	MAE[s]	MAPE	RMSE[s]	Feature set	MAE[s]	MAPE	RMSE[s]
Round 1	QL	3.6771	9.32%	4.6842	QL	2.6690	14.34%	3.6185
	NAV	1.3967	3.46%	2.3786	NAV	1.1888	6.82%	2.2235
	HW	6.0618	15.53%	7.5209	HW	3.3484	17.76%	4.4968
	ETA	5.4608	14.04%	6.7911	ETA	3.8290	19.84%	4.6471
	VD	6.4353	16.45%	8.0939	VD	4.8858	25.85%	6.0647
	FR	4.6341	11.76%	5.7694	FR	3.4285	17.85%	4.2815
Round 2	NAV+QL	1.6386	4.06%	3.0106	NAV+QL	1.2375	7.01%	2.6973
	NAV+HW	1.5842	3.89%	2.8640	NAV+HW	1.1151	6.29%	2.3574
	NAV+ETA	1.3287	3.26%	2.3469	NAV+ETA	0.5205	2.95%	1.5097
	NAV+VD	1.6177	3.97%	2.9335	NAV+VD	1.3257	7.64%	2.8670
	NAV+FR	1.5162	3.74%	2.5721	NAV+FR	1.1948	6.86%	2.2860
Round 3	NAV+ETA+QL	1.3959	3.46%	2.5778	NAV+ETA+QL	0.5513	3.10%	1.7964
	NAV+ETA+HW	1.4065	3.48%	2.5982	NAV+ETA+HW	0.5398	3.07%	1.7040
	NAV+ETA+VD	1.3977	3.44%	2.5750	NAV+ETA+VD	0.5404	3.02%	1.7978
	NAV+ETA+FR	1.4059	3.49%	2.4968	NAV+ETA+FR	0.5398	3.04%	1.6367

Figure 3.6 shows the effectiveness of the trained surrogate model. The prediction by

the trained surrogate model and the ground truth generated by I-SIG are compared in the figure. The color depth represents the density of the data. The majority of the data lie on or near the 45-degree line, indicating that the surrogate model provides a good prediction accuracy. The prediction of the lag phase has a relatively greater error because it is estimated indirectly from the barrier and the lead phase.

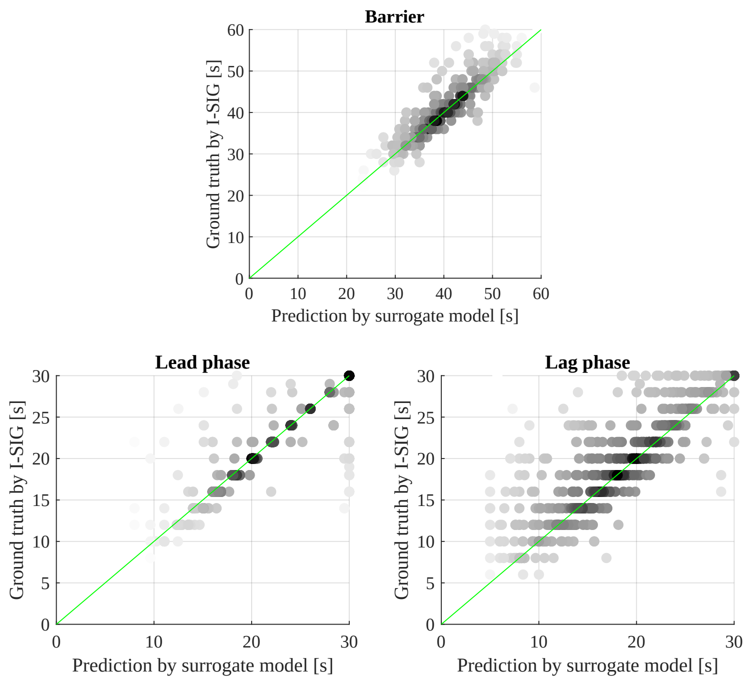


Figure 3.6: Comparison between I-SIG and the trained surrogate model

3.5.4 Evaluating the Impact of Falsified Data Attacks

To evaluate the impact of cyber attacks, attacks are launched based on the assumed attack model. In Section 3.5.3, two critical features have been identified. Therefore, two attack cases are considered, with either ETA or NAV being the targeted critical feature. The prediction by the surrogate model (i.e., the predicted signal timing plan) includes the green time of the two lead phases and two lag phases in the current barrier, and is denoted as $\boldsymbol{f}(\boldsymbol{X}) = [g_{\mathrm{d1}}, g_{\mathrm{d2}}, g_{\mathrm{g1}}, g_{\mathrm{g2}}]^{\mathsf{T}}$.

In the first case, the attacker is assumed to alter ETA by manipulating the location and speed data in the falsified trajectory. The attacker solves the following problem (**P3**) to maximize the dissimilarity between the pseudo-optimal timing plan and the timing plan under attack.

P3:

$$\begin{aligned}
\max \quad & \|\boldsymbol{f}(\boldsymbol{X_o}) - \boldsymbol{f}(\boldsymbol{X_a})\|_2 && (3.5.1)\\
\text{s.t.} \quad & \boldsymbol{X_o} = [t_{\mathrm{d1}}, t_{\mathrm{d2}}, t_{\mathrm{g1}}, t_{\mathrm{g2}}, n_{\mathrm{d1}}, n_{\mathrm{d2}}, n_{\mathrm{g1}}, n_{\mathrm{g2}}]^{\mathsf{T}} && (3.5.2)\\
& \boldsymbol{X_a} = [t_{\mathrm{d1}} + \tau_{\mathrm{d1}}\delta_{\mathrm{d1}}, t_{\mathrm{d2}} + \tau_{\mathrm{d2}}\delta_{\mathrm{d2}}, t_{\mathrm{g1}} + \tau_{\mathrm{g1}}\delta_{\mathrm{g1}}, t_{\mathrm{g2}} + \tau_{\mathrm{g2}}\delta_{\mathrm{g2}}, && \\
& n_{\mathrm{d1}} + \delta_{\mathrm{d1}}, n_{\mathrm{d2}} + \delta_{\mathrm{d2}}, n_{\mathrm{g1}} + \delta_{\mathrm{g1}}, n_{\mathrm{g2}} + \delta_{\mathrm{g2}}]^{\mathsf{T}} && (3.5.3)\\
& t_{\mathrm{d1}} + \tau_{\mathrm{d1}}\delta_{\mathrm{d1}}, t_{\mathrm{d2}} + \tau_{\mathrm{d2}}\delta_{\mathrm{d2}} \in T_{\mathrm{lead}} && (3.5.4)\\
& t_{\mathrm{g1}} + \tau_{\mathrm{g1}}\delta_{\mathrm{g1}}, t_{\mathrm{g2}} + \tau_{\mathrm{g2}}\delta_{\mathrm{g2}} \in T_{\mathrm{lag}} && (3.5.5)\\
& \delta_{\mathrm{d1}} + \delta_{\mathrm{d2}} + \delta_{\mathrm{g1}} + \delta_{\mathrm{g2}} = 1 && (3.5.6)\\
& \tau_{\mathrm{d1}}, \tau_{\mathrm{d2}}, \tau_{\mathrm{g1}}, \tau_{\mathrm{g2}} \geq 0 && (3.5.7)\\
& \delta_{\mathrm{d1}}, \delta_{\mathrm{d2}}, \delta_{\mathrm{g1}}, \delta_{\mathrm{g2}} \in \{0, 1\} && (3.5.8)\\
\text{Decision variables} \quad & \tau_{\mathrm{d1}}, \tau_{\mathrm{d2}}, \tau_{\mathrm{g1}}, \tau_{\mathrm{g2}}, \delta_{\mathrm{d1}}, \delta_{\mathrm{d2}}, \delta_{\mathrm{g1}}, \delta_{\mathrm{g2}} && (3.5.9)
\end{aligned}$$

The observed critical features $\boldsymbol{X_o}$ include the ETAs of the two lead phases $(t_{\text{d1}}, t_{\text{d2}})$ and two lags phases $(t_{\text{g1}}, t_{\text{g2}})$ as well as the NAVs of the two lead phases $(n_{\text{d1}}, n_{\text{d2}})$ and two lag phases $(n_{\text{g1}}, n_{\text{g2}})$ (Equation 3.5.2). Similarly, the altered critical features $\boldsymbol{X_a}$ also include corresponding ETAs and NAVs (Equation 3.5.3). The binary variables $\delta_{\text{d1}}, \delta_{\text{d2}}, \delta_{\text{g1}}, \delta_{\text{g2}}$ indicate the phase to which the falsified trajectory is injected (Equation 3.5.8). It is assumed that the attacker can only inject one falsified trajectory per attack (Equation 3.5.6). T_{lead} and T_{lag} are the sets of candidate ETAs for the lead phase and lag phase (Equation 3.5.4 and 3.5.5). T_{lead} and T_{lag} can be obtained from historical data. Equation 3.5.7 indicates that the attacker intends to launch attacks by increasing the ETA.

In the second case, the attacker is assumed to alter NAV by injecting falsified trajectories to different phases. Denote $\delta_{\text{d1}}, \delta_{\text{d2}}, \delta_{\text{g1}}, \delta_{\text{g2}}$ to be the number of falsified trajectories injected to each phase. The attacker is assumed to have a budget limit $B = 10$ (Equation 3.5.13), i.e., the maximum number of falsified trajectories. Similar to the previous case, the attacker solves the following problem (**P4**) to maximizes the dissimilarity.

P4:

$$\max \quad \|\boldsymbol{f}(\boldsymbol{X_o}) - \boldsymbol{f}(\boldsymbol{X_a})\|_2 \tag{3.5.10}$$

$$\text{s.t.} \quad \boldsymbol{X_o} = [t_{\text{d1}}, t_{\text{d2}}, t_{\text{g1}}, t_{\text{g2}}, n_{\text{d1}}, n_{\text{d2}}, n_{\text{g1}}, n_{\text{g2}}]^\mathsf{T} \tag{3.5.11}$$

$$\boldsymbol{X_a} = [t_{\text{d1}}, t_{\text{d2}}, t_{\text{g1}}, t_{\text{g2}},$$

$$n_{\text{d1}} + \delta_{\text{d1}}, n_{\text{d2}} + \delta_{\text{d2}}, n_{\text{g1}} + \delta_{\text{g1}}, n_{\text{g2}} + \delta_{\text{g2}}]^\mathsf{T} \tag{3.5.12}$$

$$\delta_{\text{d1}} + \delta_{\text{d2}} + \delta_{\text{g1}} + \delta_{\text{g2}} \leq B \tag{3.5.13}$$

$$\delta_{\text{d1}}, \delta_{\text{d2}}, \delta_{\text{g1}}, \delta_{\text{g2}} \in \mathbb{Z}^+ \tag{3.5.14}$$

$$\text{Decision variables} \quad \delta_{\text{d1}}, \delta_{\text{d2}}, \delta_{\text{g1}}, \delta_{\text{g2}} \tag{3.5.15}$$

Four simulation experiments are conducted to assess the impact of cyber attacks. Each experiment lasts for 5 hours, with the exact same traffic demand and vehicle arrival patterns. The total delay for each experiment is shown in Figure 3.7.

In Experiment I, the original I-SIG system operates normally without being attacked. This scenario serves as the benchmark for all the other scenarios because it has the lowest total delay.

In Experiment II, the trained surrogate model is used to generate signal timing plans and control the traffic signal. However, no falsified CV data are injected into the system. As evidenced by a small increase of 1.5%, the total delay is very close to the benchmark. This indicates that the trained surrogate model could effectively mimic the actual signal

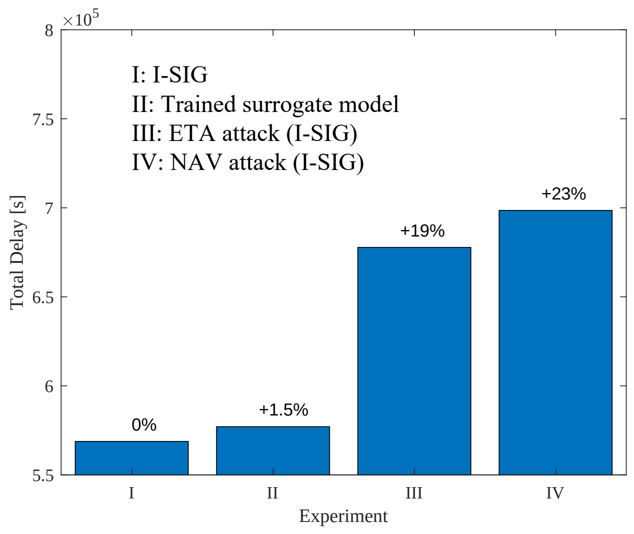


Figure 3.7: Total delay for each experiment

control model.

In Experiment III, the attacker attacks I-SIG based on the feature ETA. It is worth noting that only one falsified CV is inserted into the system per attack and the attacker is constrained by limited information (unknown control model). The total delay increases by 19%. This implies that the attacker can cause significant damage to the CV-TSC system even though the exact control model is unknown.

In Experiment IV, the attacker attacks I-SIG based on the feature NAV. Multiple falsified CVs are inserted into the system per attack. The total delay increases by 23%. When sufficient resources are given, the attacker can cause even more damage to the system.

These experiment results show that the attacker can learn the signal control model with a surrogate model. The attack model is simple yet effective. It is possible to attack a CV-TSC system without knowing the exact control model. The falsified data attacks can result in severe consequences with a limited budget.

3.6 Chapter Summary

This chapter investigated how an attacker can launch real-time falsified data attacks in a realistic setting. Compared to prior research, this study considered a "black-box" attack scenario in which the control model of the CV-TSC system was unavailable to the attacker. It was assumed that the attacker could learn the signal control model using a surrogate model and then identify critical traffic features. Using the learned model, the attacker could predict the signal timing plan based on identified critical traffic features. The attacker then formulated a mathematical problem to generate falsified trajectories, with the aim of altering the values of critical traffic features. Signal control decisions were affected as a result. The attacker was assumed to identify the "optimal" values of critical features by maximizing dissimilarity between the pseudo-optimal timing plan and the resultant signal timing plan under attack.

A comprehensive case study was performed with I-SIG as the selected CV-TSC system. Results showed that the surrogate model could effectively mimic the actual I-SIG control model, which was sensitive to two critical traffic features: the estimated time of arrival (ETA) and number of approaching vehicles (NAV). Therefore, two types of attacks could be launched based on these features. Simulation experiments revealed that the total delay increased by 19% and 23%, respectively. This indicated that even though the control model was unknown, an attacker could still severely damage the CV-TSC system.

Note that it is possible that the attacker may not be able to identify critical traffic features or the surrogate model may not work well. In this case, the attacker will not be able to launch effective black-box attacks. Moreover, in this study, **P1** and **P2** are solved separately when launching online attacks. **P1** and **P2** can potentially be combined into one model.

The study in this chapter further confirms the impacts of cyber attacks on the CV-TSC system. Even in a restricted setting (i.e., black box), an attacker can create excessive delay and hinder system performance significantly. A defense solution that can be used to detect such attacks and protect the system is vital. The defense part of the cyber security problem is explored in the next chapter.

CHAPTER 4

Defense Strategy: A Data-Driven Method to Identify Falsified Trajectories

4.1 Introduction

The previous two chapters provided evidence that falsified trajectories can significantly damage CV-TSC systems. Besides traffic signal operation, falsified trajectories may harm other trajectory-based applications as well, such as traffic state estimation. Input vehicle trajectories must be authentic to secure the benefits of trajectory-based applications. It is therefore imperative to protect the system by identifying and filtering out falsified vehicle trajectories, hence the focus of this chapter.

One straightforward approach to identifying falsified trajectories involves using infrastructure-based sensors (e.g., loop detectors) to validate unknown trajectories. For instance, Canepa and Claudel (2013a,b) formulated a mixed-integer linear feasibility problem to detect falsified trajectories. Traffic states were modeled using the Lighthill-

Whitham-Richards model (Lighthill and Whitham, 1955; Richards, 1956). Detector data provided initial and boundary conditions for this model. The information (e.g., average speed) obtained from falsified trajectories could influence traffic state estimation, rendering the original mixed-integer linear problem infeasible. Shoukry et al. (2018) also used legacy loop detectors to estimate macroscopic traffic states. A set of honest vehicles were then identified, whose velocity values were consistent with macroscopic traffic states. However, the applicability of these methods is limited by the need to acquire data from infrastructure-based sensors.

Another approach to identifying falsified trajectories is to model the problem as an abnormal (outlier) trajectory identification problem. An abnormal trajectory is distinct from other trajectories in terms of a distance metric (Zheng, 2015). Research identifying abnormal trajectories has most often focused on the network-level behavior, namely route choice (Zhang et al., 2011; Chen et al., 2013; Zhu et al., 2015, 2017). Such studies normally design a routing-related score to identify abnormal trajectories. Yet one major drawback of these methods is that they cannot be directly applied at the intersection level, which is the context of this dissertation.

To address the research gaps identified in prior literature, this chapter proposes a data-driven method for identifying falsified trajectories. A trajectory embedding model is developed that generates vector representations of driving behaviors. Using these vectors, the proposed method computes the similarity between trajectories and therefore identifies abnormal (falsified) ones. This method can be applied at the intersection level and does not require data from infrastructure-based sensors. Therefore, the method can be used to protect CV-TSC systems from falsified data attacks.

The rest of this chapter is organized as follows. Section 4.2 describes the falsified trajectory identification problem and provides an overview of the proposed method. Section 4.3 details the trajectory embedding model to generate vector representations. Section 4.4 explains how to identify falsified trajectories using the vector representations, including computing the similarity between trajectories and a hierarchical clustering algorithm to identify abnormal (falsified) trajectories. Numerical examples are presented in Section 4.5 to demonstrate the effectiveness of the proposed method. Finally, Section 4.6 summarizes this chapter.

4.2 Problem Description and Methodology Overview

A time-space diagram at a signalized intersection is shown in Figure 4.1. The blue solid curves represent trajectories of CVs (i.e., observable by infrastructure) and the blue dashed curves represent trajectories of non-CVs (i.e., not observable by infrastructure). The red curve represents a falsified trajectory. The resolution of the trajectories is 10 Hz (i.e., one trajectory data point for every 0.1 second). In this example, the falsified trajectory behaves differently from the normal ones because it intentionally slows down although the front vehicle is still far away. It is assumed that at the time of interest (TOI), a CV-TSC system utilizes received trajectories (the blue solid trajectory and the red trajectory) as input data for traffic signal optimization. The goal of the defense strategy is to identify the falsified trajectory for each signal phase before TOI.

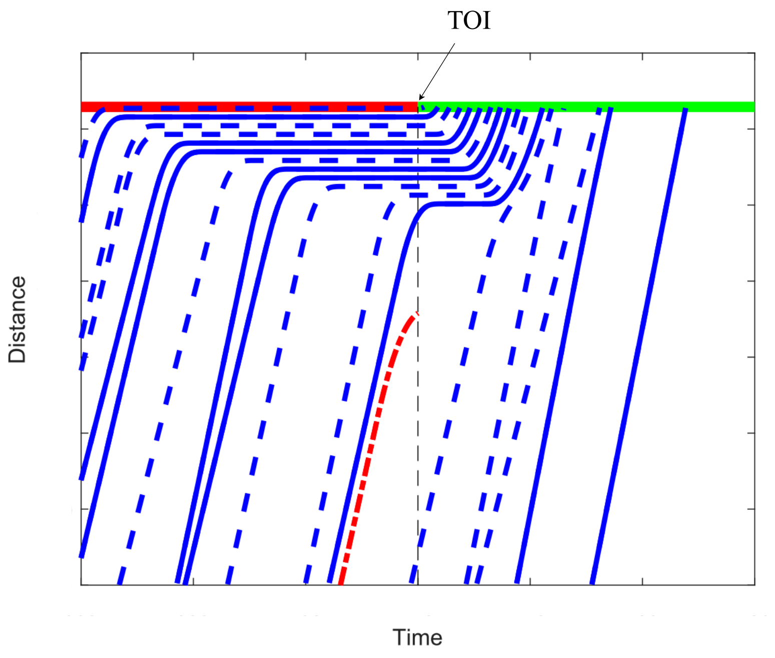


Figure 4.1: Illustration of the problem of identifying the falsified trajectory

The falsified trajectory identification method proposed in this chapter is an extension of the trajectory-based hierarchical defense (TBHD) framework (Wong et al., 2019). The TBHD framework aims at detecting falsified trajectory data (i.e., BSMs) from compromised CVs. The TBHD framework consists of three levels of defenses. Level 1 defense is a pointwise checking that checks if each element in received BSMs fall within its feasible range. It checks the location, speed, acceleration, and heading of each trajectory data point and makes sure that these elements make physical sense. Level 2 defense is a multiple-point checking that checks if consecutive BSMs of one CV obey the laws of physics. It checks the location, speed, and acceleration of each trajectory data point based on equations of motion and the definitional equations of speed and acceleration. Level 3 is a multiple-trajectory checking that checks if two CV trajectories conflict with each other. The core idea is that if a trajectory conflicts with multiple trajectories, then it is considered falsified. The three levels of defenses in the TBHD are fairly simple. An attacker can easily defeat these defenses. For example, the falsified trajectory in Figure 4.1, is generated by solving **P2** in Section 3.4. The three levels of defenses are treated as constraints in **P2**. As a result, the falsified trajectory can successfully pass the three levels of defenses.

The proposed method provides another level of defense by checking the behavior of trajectories. The overview of this method is shown in Figure 4.2. This method includes two key parts: trajectory embedding and abnormal trajectory identification. A trajectory is composed of multiple trajectory data points. Each trajectory data point reveals driving behavior (i.e., the choice of range, range rate, and speed). Inspired by a word embedding model from the Natural Language Processing (NLP) community, a trajectory embedding model is developed. Each trajectory data point is converted into a word via trajectory pre-processing. The trajectory embedding model then encodes the word into a vector. The trajectory embedding model is a simple neural network with a single hidden layer. It is trained on positive samples (correct context-target word pairs) and negative samples (incorrect context-target word pairs). Each context word is represented as a distinct input to the neural network. The output is the probability of a target word (for example, Detroit-Michigan as the context-target word pair). The training of the neural network is essentially to estimate the co-occurring probability of any pair of words. The weights of the hidden layer are the vector representation of a word. After training, the vector representations of similar words will end up near in the vector space, i.e., the neural network model implies that words that occur in similar contexts would have similar vector representations (similar hidden layers). In trajectory embedding, trajectory data points with similar driving behavior have similar vector representations. The vector representations enable the computation of trajectory similarity. A falsified trajectory is identified based on its similarity (distance) to other trajectories. The distance between two trajectory data points can be calculated using the Euclidean distance between the two vectors. The similarity between the two trajectories is then calculated as the average distance over a time window. A similarity matrix is then obtained by computing the similarity between all pairs of trajectories. Next, a hierarchical clustering algorithm is adopted to merge similar trajectories into clusters. A predefined threshold is used to terminate the merging process. Trajectories in the largest cluster are marked as normal, whereas other trajectories are considered abnormal (i.e., falsified).

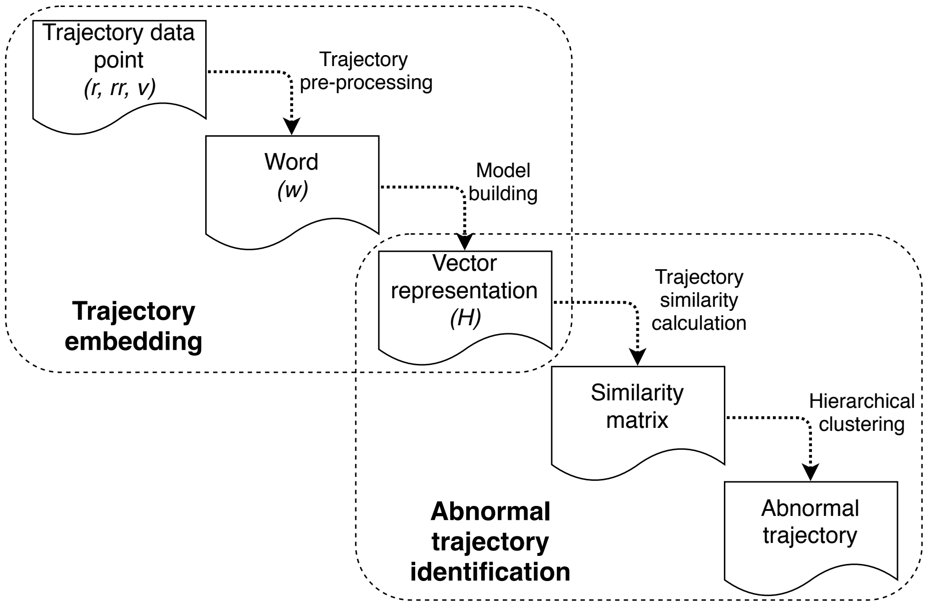


Figure 4.2: Overview of the falsified trajectory identification method

The proposed method is grounded on two basic assumptions: (1) falsified trajectories only account for a small proportion of the vehicle population. This means that the attacker has limited resources and most trajectories are normal; and (2) falsified trajectories behave differently from normal trajectories. As discussed in Chapter 3, falsified trajectories are generated with a certain attack goal (e.g., to alter the values of critical traffic features). Therefore, falsified trajectories may demonstrate driving behavior unique from normal trajectories. This behavioral difference is also supported by Chen et al. (2018), whose study showed that an attacker can attack I-SIG by changing speed and location data elements in the trajectory and thus manipulating arrival time or queue length.

The logic of the proposed method is explained below. Normal trajectories share similar vector representations because of similar driving behavior. Falsified trajectories, however, have dissimilar vector representations due to the behavioral difference. The vector representations are used to compute the similarity between trajectories. As a result, falsified trajectories are dissimilar to normal trajectories. Based on the similarity, the hierarchical clustering algorithm will merge normal trajectories into one single cluster. The falsified trajectories are then identified.

4.3 Trajectory Embedding

The trajectory embedding model is inspired by a word embedding model called word2vec (Mikolov et al., 2013a,b,c), a popular tool primarily used in the NLP community. Word2vec uses real-valued vectors to represent words. Each word is mapped into a low dimensional vector (by "low" the dimension of the vector is compared to the size of the vocabulary of words). Words with similar meanings end up with similar vector representations. The vector representation carries semantic information and can be used for various downstream NLP applications. Just like a sentence is composed of multiple words, a trajectory is composed of multiple trajectory data points. Each trajectory data point can be considered an analogue to a word in a sentence. The trajectory embedding model generates vector representations of trajectory data points. Similarly, it is expected that trajectory data points with similar driving behavior have similar vector representations. The vector representations make it possible to to compute the similarity between trajectories. This section includes two parts: trajectory pre-processing and embedding model building. The first part describes how to convert a trajectory data point into a word. The second part explains how to train the model and generate vector representations of words (i.e., trajectory data points).

4.3.1 Trajectory Pre-Processing

A trajectory is composed of a series of trajectory data points. Each trajectory data point contains the location, speed of the vehicle with a timestamp. For each trajectory data point, car-following information such as range (the space gap between the host CV and its leading CV) and range rate (the speed difference between the host CV and its leading CV) can be derived directly based on the trajectories of the host CV and its leading CV. Note that when the CV market penetration rate is not 100%, the leading CV is not necessarily the immediate leading vehicle, as shown in Figure 4.1. In other words, there may be non-CVs in between the CVs.

Range, range rate, and speed are considered as driving behavior in this study. The driving behavior is then mapped into a three-letter word using a mapping function $g(r, rr, v)$. The three letters correspond to the values of range r, range rate rr, and speed v, respectively. The mapping function $g(r, rr, v)$ discretizes the parameter space. In this study, range r has 26 possible intervals with 2 meters per interval: $[0, 2\text{m})$, $[2\text{m}, 4\text{m})$, ..., $[48\text{m}, 50\text{m})$,$[50\text{m}, \infty)$. Range rate rr has 13 possible intervals with 2 m/s per interval: $(-\infty, -18\text{m/s})$, $[-18\text{m/s}, -16\text{m/s})$,...,$[2\text{m/s}, 4\text{m/s})$, $[4\text{m/s}, \infty)$. Speed v has 11 possible intervals with 2 m/s per interval: $[0, 2\text{m/s})$, $[2\text{m/s}, 4\text{m/s})$,...,$[18\text{m/s}, 20\text{m/s})$, $[20\text{m/s}, \infty)$.

The upper and lower boundaries of each parameter are obtained from simulation data. Each interval corresponds to a letter alphabetically. For example, driving behavior $(r = 1\text{m/s}, rr = -17\text{m/s}, v = 5\text{m/s})$ corresponds to the word "abc". With this mapping function $g(r, rr, s)$, any trajectory data point can be mapped into a word.

4.3.2 Embedding Model Building

The embedding model is adapted from the simplest version of the word2vec model (one input word and one output word) with negative sampling approach. For a detailed explanation of the word2vec model, please refer to Goldberg and Levy (2014) and Rong (2014). The embedding model generates vector representations of words based on the distribution of word co-occurrences in a training dataset. As shown in Figure 4.3, the embedding model is a neural network with three layers: an input layer, an output layer, and a hidden layer. The model has an internal prediction task: predicting a target word given a context word. The model parameters are updated through training. However, the prediction task itself is not the goal of this model. The goal is to learn the weights of the hidden layer. Eventually, each word will have its unique weights of the hidden layer and these weights are the vector representation of the word. The vector representation can be considered as features that describe this word.

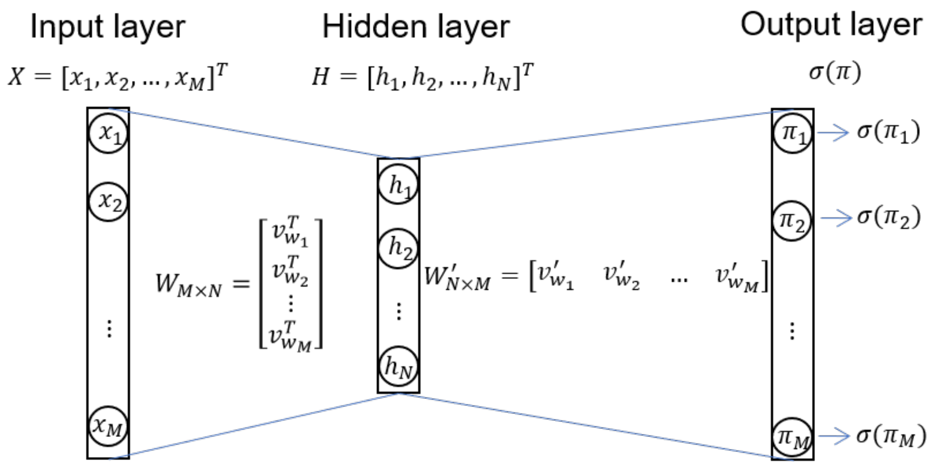

Input layer
$X = [x_1, x_2, \dots, x_M]^T$
Hidden layer
$H = [h_1, h_2, \dots, h_N]^T$
Output layer
$\sigma(\pi)$
x_1
x_2
x_M
$W_{M\times N} = \begin{bmatrix} v_{w_1}^T \\ v_{w_2}^T \\ \vdots \\ v_{w_M}^T \end{bmatrix}$
h_1
h_2
h_N
$W'_{N\times M} = [v'_{w_1} \quad v'_{w_2} \quad \dots \quad v'_{w_M}]$
π_1
π_2
π_M
$\sigma(\pi_1)$
$\sigma(\pi_2)$
$\sigma(\pi_M)$

Figure 4.3: Illustration of the embedding model

4.3.2.1 Model Description

Denote V as the vocabulary (i.e., unique words) and M as the number of unique words (i.e., the size of the vocabulary). The input layer is a one-hot encoded vector with dimension M. Given a context word $w_i \in V$ (here i indicates its location in the vocabulary), the input to the embedding model is $X_{w_i} = [0, 0, ...0, 1, 0., , , .0]^\mathsf{T}$, where only i-th element is 1 and all other elements are 0. The weight between the input layer and the hidden layer is a $M \times N$ matrix denoted as W. Row i of W is denoted as $v_{w_i}^\mathsf{T}$ with dimension N. The dimension of the hidden layer is N, which is a predefined value (N is significantly smaller than M). In this study, $N = 20$. Given a word $w_i \in V$, the hidden layer can be computed as $H = W^\mathsf{T} X_{w_i} = v_{w_i}$. The weight between the hidden layer and output layer is a $N \times M$ matrix denoted as W' (W' is different from W). Column i of W' is denoted as v'_{w_i} with dimension N. The dimension of the output layer is M. For each output word w_j in the vocabulary, the score π_j is computed as $\pi_j = {v'_{w_j}}^\mathsf{T} H = {v'_{w_j}}^\mathsf{T} v_{w_i}$. v_{w_i} is called the input vector of the word w_i and ${v'}_{w_j}$ the output vector of the word w_j. The probability of predicting word w_j is given by a logistic function $\sigma(\pi_j) = \sigma({v'}_{w_j}{}^\intercal v_{w_i}) = \frac{1}{1+\exp\left(-{v'}_{w_j}{}^\intercal v_{w_i}\right)}$.

Denote w_{i^*} as the target word of w_i, i.e., the positive sample. Denote $D(w_i)$ as the set of negative samples. $D(w_i)$ contains k "wrong" words (i.e., any word other than the target word) that are randomly drawn from the vocabulary V (thus the name negative sampling). In this study, k is set to 5, the recommended value for a small training dataset (Mikolov et al., 2013b). The loss function is defined in Equation 4.3.1.

$$E = -\log \sigma({v'}_{w_{i^*}}{}^\intercal v_{w_i}) - \sum_{w_j \in D(w_i)} \log \sigma(-{v'}_{w_j}{}^\intercal v_{w_i}) \tag{4.3.1}$$

Take derivative of E with respect to ${v'}_{w_j}$:

$$\begin{aligned} \frac{\partial E}{\partial {v'}_{w_j}} &= \frac{\partial E}{\partial({v'}_{w_j}{}^\intercal v_{w_i})} \frac{\partial({v'}_{w_j}{}^\intercal v_{w_i})}{{v'}_{w_j}} \\ &= [\sigma({v'}_{w_j}{}^\intercal v_{w_i}) - t(w_j)] \frac{\partial({v'}_{w_j}{}^\intercal v_{w_i})}{{v'}_{w_j}} \\ &= [\sigma({v'}_{w_j}{}^\intercal v_{w_i}) - t(w_j)] v_{w_i} \end{aligned} \tag{4.3.2}$$

where

$$t(w_j)) \begin{cases} 1, & w_j = w_{i^*} \\ 0, & w_j \in D(w_i) \end{cases} \tag{4.3.3}$$

Therefore, using stochastic gradient descent, the update rule for the weight ${v'}_{w_j}$ in matrix W' is:

$$v'_{w_j}{}^{\text{(new)}} = v'_{w_j}{}^{\text{(old)}} - \eta \frac{\partial E}{\partial v'_{w_j}} \tag{4.3.4}$$

where $\eta > 0$ is the learning rate.

Take derivative of E with respect to v_{w_i}:

$$\begin{aligned}
\frac{\partial E}{\partial v_{w_i}} &= \sum_{w_j \in \{w_{i^*}\} \cup D(w_i)} \frac{\partial E}{\partial ({v'_{w_j}}^{\mathsf{T}} v_{w_i})} \frac{\partial ({v'_{w_j}}^{\mathsf{T}} v_{w_i})}{v_{w_i}} \\
&= \sum_{w_j \in \{w_{i^*}\} \cup D(w_i)} [\sigma({v'_{w_j}}^{\mathsf{T}} v_{w_i}) - t(w_j)] \frac{\partial ({v'_{w_j}}^{\mathsf{T}} v_{w_i})}{v_{w_i}} \\
&= \sum_{w_j \in \{w_{i^*}\} \cup D(w_i)} [\sigma({v'_{w_j}}^{\mathsf{T}} v_{w_i}) - t(w_j)] v'_{w_j}
\end{aligned} \tag{4.3.5}$$

The corresponding update rule for the weight v_{w_i} in matrix W is

$$v_{w_i}{}^{\text{(new)}} = v_{w_i}{}^{\text{(old)}} - \eta \frac{\partial E}{\partial v_{w_i}} \tag{4.3.6}$$

For each input word w_i, Equation 4.3.4 is applied to $w_j \in \{w_{i^*}\} \cup D(w_i)$ and update the corresponding v'_{w_j} in matrix W' and Equation 4.3.6 is applied to update v_{w_i} in matrix W.

For each iteration, one goes through all the positive and negative samples from a training dataset and updates the weight matrix W and W' based on Equation 4.3.4 and 4.3.6.

Given an arbitrary word $w_i \in V$, its vector representation is the vector of its hidden layer H. That is to say, the vector representation of word w_i is $H = v_{w_i}$.

Following Rong (2014), the logic behind the embedding model is explained below. $\sigma({v'_{w_j}}^{\mathsf{T}} v_{w_i})$ is the probability of the correct prediction (takes value between 0 and 1). $t(w_j)$ is the expected output (takes value of either 0 or 1). If $\sigma({v'_{w_j}}^{\mathsf{T}} v_{w_i}) > t(w_j)$, the word w_j is not the correct target word of w_i. Equation 4.3.4 subtracts a proportion of v_{w_i} from v'_{w_j}, and moves v'_{w_j} farther away from v_{w_i}. The step size of the movement is determined by the prediction error $\sigma({v'_{w_j}}^{\mathsf{T}} v_{w_i}) - t(w_j)$. Similarly, Equation 4.3.6 subtracts a proportion of v'_{w_j} from v_{w_i}, making v_{w_i} farther away from v'_{w_j}. If $\sigma({v'_{w_j}}^{\mathsf{T}} v_{w_i}) < t(w_j)$, the word w_j is the correct target word of w_i. Equation 4.3.4 adds a proportion of v_{w_i} to v'_{w_j}, and moves v'_{w_j} closer to v_{w_i}. Similarly, Equation 4.3.6 adds a proportion of v'_{w_j} to v_{w_i}, making v_{w_i} closer to v'_{w_j}. If $\sigma({v'_{w_j}}^{\mathsf{T}} v_{w_i})$ is very close to $t(w_j)$, little change will be made to v'_{w_j} in Equation 4.3.4 and v_{w_i} in Equation 4.3.6. After many iterations, the weight matrix W and W' will stabilize eventually.

4.3.2.2 Model Training

The trajectory embedding model is trained on positive and negative samples. As discussed in Section 4.3.2.1, for each positive sample, k negative samples are randomly chosen from the vocabulary. This section introduces how to construct positive samples for the trajectory embedding model. In this study, there are three types of positive samples.

- Single-trajectory (temporal dimension)

  As illustrated in Figure 4.4(a), a complete trajectory i is recorded. Each trajectory data point is converted into a word and is denoted as w_t^i, where t is the timestamp.

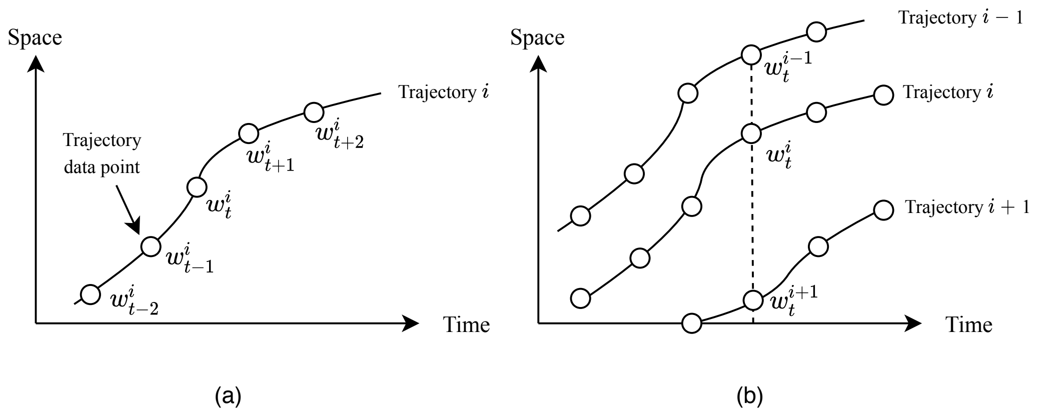


Figure 4.4: Illustration of constructing positive samples: (a) single-trajectory (temporal dimension); and (b) between-trajectory (spatial dimension)

Given a word w^i_t, its immediate next word w^i_{t+1} (i.e., next timestamp) is its positive sample. This type of positive samples ensures that adjacent trajectory data points in the temporal dimension have similar vector representations.

- Between-trajectory (spatial dimension)

  As illustrated in Figure 4.4(b), at time t, a snapshot is taken. The trajectory data points of all CVs are converted into words. Given a word w^i_t, its front word w^{i-1}_t (i.e., leading CV) and back word w^{i+1}_t (i.e., following CV) are its positive samples. This type of positive samples ensures that adjacent trajectory data points in the spatial dimension have similar vector representations.

- Related words

  Related words are defined as the words that only differ by one letter and are next to each other in the mapping function. For example, “aaa” and “aab” are related words. “aab” is the positive sample of “aaa” and “aaa” is the positive sample of "aab". This type of positive samples ensures that similar driving behavior have similar vector representations. Moreover, this type of positive samples would enumerate all possible words in the vocabulary and therefore can prevent out-of-vocabulary words (i.e., driving behavior that does not appear in a training data set but appears in a testing data set).

4.4 Abnormal Trajectory Identification

The trained trajectory embedding model provides vector representations of trajectory data. Therefore, the similarity (distance) between any pair of trajectories can be computed using the vector representations. Then a clustering algorithm is applied to group similar trajectories and identify abnormal ones.

4.4.1 Trajectory Similarity Calculation

Figure 4.5 shows two trajectories in a time-space diagram. Each trajectory is composed of multiple trajectory data points. t_s and t_e define the time window that both trajectories are received by a CV-TSC system. At any time t within this time window, the distance between two trajectory points can be calculated using the Euclidean distance of the two vector representations (i.e., $\|H_t^i - H_t^j\|_2$). Then, the similarity between the two trajectories is defined as the average distance over the time window, as shown in Equation 4.4.1.

$$d(i,j) = \frac{1}{10(t_e - t_s) + 1} \sum_{t=t_s}^{t_e} \|H_t^i - H_t^j\|_2 \tag{4.4.1}$$

where H_t^i and H_t^j are the vector representations for trajectory i and j at time t; $10(t_e - t_s) + 1$ denotes the number of data points in the time window.

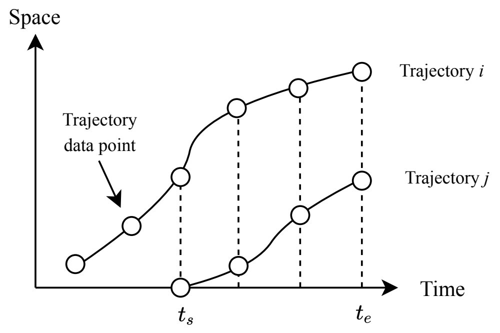


Figure 4.5: Illustration of computing the similarity between two trajectories

4.4.2 Classification Using Hierarchical Clustering

Hierarchical clustering is an unsupervised learning method in machine learning. It groups similar objects to clusters such that objects in the cluster are similar to each other than the objects in other clusters. Hierarchical clustering is a popular method for clustering DNA sequences.

In this study, hierarchical clustering is adopted to identify abnormal trajectories. The pseudo code is shown in Algorithm 2. This study uses a bottom-up approach. Each trajectory is treated as one cluster in the beginning. The similarity between clusters c_p and c_q is calculated as the average distance between all pairs of trajectories in the two clusters, i.e., $D(c_p, c_q) = \frac{1}{|c_p||c_q|} \sum\limits_{i \in c_p, j \in c_q} d(i, j)$. For each iteration, the most similar clusters are merged together to form one single cluster. The merging stops when the clusters are not close to each other (i.e., the similarity exceeds a predefined threshold ϵ) or all trajectories have been merged into the same cluster. Finally, the trajectories in the largest cluster are marked as normal; otherwise abnormal.

Algorithm 2: Hierarchical clustering

```
Consider each trajectory as one cluster;
Compute the minimum distance between any pair of clusters
 $D_{min} = \min\limits_{c_p, c_q \in C} D(c_p, c_q)$, where $C$ is the set of clusters ;
while $D_{min} \leq \epsilon$ & $|C| > 1$ do
    Merge cluster $c_p^*$ and $c_q^*$ into one cluster, where $D(c_p^*, c_q^*) = D_{min}$ ;
    if $|C| > 1$ then
        Compute $D_{min} = \min\limits_{c_p, c_q \in C} D(c_p, c_q)$;
    else
        break while;
    end
end
Result: The trajectories in the largest cluster are marked as normal; otherwise
        abnormal
```

Figure 4.6 is a dendrogram that shows how hierarchical clustering is performed from the bottom (every trajectory is one cluster) to the top (all trajectories merge into one cluster). The threshold in hierarchical clustering is set to 6. Trajectories 2 to 10 eventually merge and form one cluster, while trajectory 1 forms the second cluster. In this example, trajectory 1 is marked as abnormal (i.e., falsified).

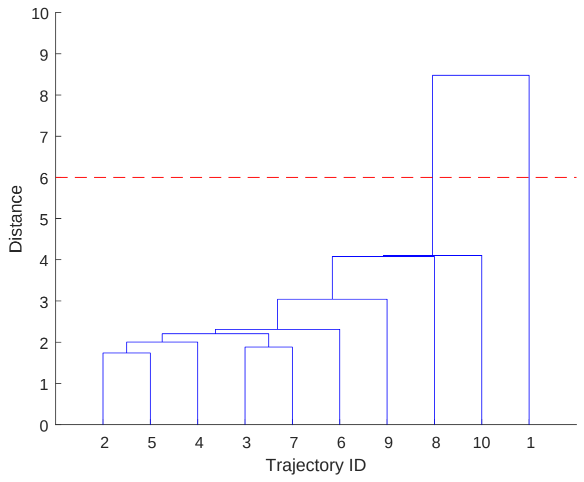


Figure 4.6: A dendrogram that shows the hierarchical clustering results

4.5 Numerical Examples

4.5.1 Experiment Setup

Experiments are conducted to demonstrate the effectiveness of the proposed method. VISSIM (Fellendorf and Vortisch, 2010), a commercial microscopic traffic simulator, is utilized as the simulation environment. In order to ensure heterogeneity in driving behavior, the simulation uses stochastic values for safety distance, desired acceleration, and desired deceleration. The free-flow speed follows a uniform distribution, with a lower bound of 50 km/h (13.89 m/s) and an upper bound of 70 km/h (19.44 m/s). A simple one-lane road with a traffic light at the end of the road is modeled in the simulation. The traffic signal implements a fixed-time signal timing plan, with 60 seconds of green time and 60 seconds of red time for every cycle. TOI is defined as the time that the traffic signal switches from red to green. The traffic volume is set to 400 vehicles per hour. The simulation lasts for 10 hours. 300 signal cycles of vehicle trajectory data are recorded. The simulation resolution is 10 Hz, where the trajectory data is recorded every 0.1 second. These 10 hours of data are used as the corpus for training the trajectory embedding model. In the same way, another 10 hours of data are generated. For each cycle, one falsified trajectory is inserted between free-flow traffic and queueing traffic (generated 2 seconds after the generation of the last CV in the queueing traffic) because the falsified trajectory is normally not in free-flow state or queueing state. The falsified trajectory is generated offline (given the full trajectory of the leading and the following vehicle) by solving **P2** in Section 3.4. The time displacement and space displacement in **P2** are set 1.5 seconds and 6.2 meters. The safety distance is 6.2 meters. The free-flow speed is 16.67 m/s (60 km/h). The minimum and maximum acceleration rate are -2 m/s^2 and 2 m/s^2. The initial speed of the falsified trajectory is the same as the free-flow speed and the initial position is at the boundary of the approach. Recall that one constraint in **P2** is the attack goal. A total of 6 cases are considered with three attack goals (ETA attack, phantom queue attack, and without a goal) and two CV penetration rates (100% and 50%). Each case includes 300 signal cycles. The ETA attack and phantom queue attack correspond to the findings in Chen et al. (2018), which finds that an attacker can generate falsified CV data to fool the signal control system by either manipulating arrival time or queue length (note that phantom queue attack is effective only when the penetration rate is less than 100%).

For the ETA attack, the mathematical formulation of the attack goal (Equation 3.4.9

in **P2**) is expressed in the following equations:

$$v(t_o) > v_{\text{min}} \tag{4.5.1}$$

$$\frac{d_b - d(t_o)}{v(t_o)} = \text{ETA} \tag{4.5.2}$$

Equation 4.5.1 states that at TOI t_o, the speed of the falsified trajectory should be greater than a minimum speed v_{min} (in this way the signal controller would think this vehicle is still moving). Equation 4.5.2 states that ETA is defined as the distance to stop bar $d_b - d(t_o)$ divided by speed $v(t_o)$. The ETA attack would trick the signal controller into believing that there is a late arrival vehicle that needs to be served.

For the phantom queue attack, the mathematical formulation of the attack goal is expressed in the following equations:

$$v(t_o) = 0 \tag{4.5.3}$$

$$(d_b - d(t_o)) \times k_j = \text{QUEUE} \tag{4.5.4}$$

Equation 4.5.3 states that at TOI t_o, the speed of the falsified vehicle should be 0. Equation 4.5.4 states that the number of queued vehicles QUEUE is equal to the product of the distance to stop bar $d_b - d(t_o)$ and the jam density k_j (161 veh/km). The phantom queue attack would trick the signal controller into believing that there is a long vehicle queue (especially when the penetration rate is not 100% and a signal controller normally uses the last stopped CV to estimate queue length).

4.5.2 Trajectory Embedding Model Training

Gensim (Řehůřek and Sojka, 2010), a python package for word embedding, is used for training the trajectory embedding model. Two trajectory embedding models are trained based on different penetration rates (PR). The delta loss (the difference of loss between two iterations) is shown in Figure 4.7. When the penetration rate is 100%, the training converges near 1000 iterations. When the penetration rate is 50%, the training converges near 2300 iterations. When the penetration rate is low, it takes longer time to converge. This is because range and range rate are more diverse.

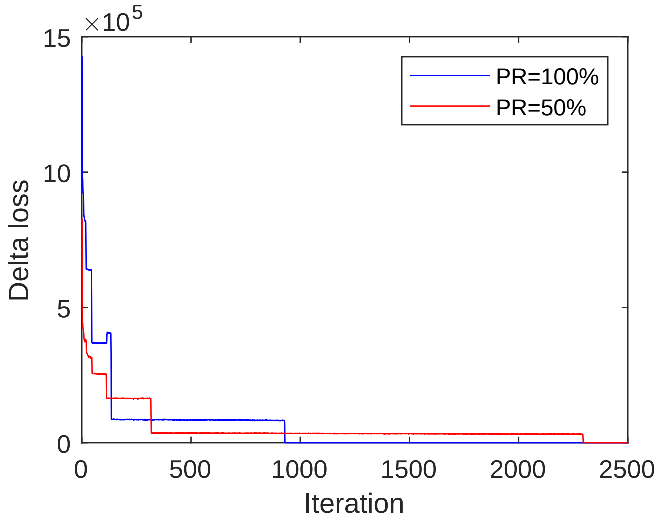


Figure 4.7: Delta loss for each iteration

Table 4.1 shows a few examples of the trajectory embedding model (PR=100%). Euclidean distance is used to identify similar words (i.e., $\|H(w_i) - h(H_j)\|_2$, where $h(H_i)$ and $h(H_j)$ are the vector representations of words w_i and w_j). A small Euclidean distance indicates that the two words are similar. For example, the top 2 similar words to "aka" are “aja” and “bja”. All three words represent queueing traffic state. The top 2 similar words to “zkh” are “ukh” and “zji”. Similarly, all three words represent free-flow traffic state.

Table 4.1: Similar words according to Euclidean distance

Group	Euclidean Distance	Range [m]	Range rate [m/s]	Speed [m/s]	Word
1	-	[0,2)	[0,2)	[0,2]	aka
	0.80	[0,2)	[-2,0]	[0,2]	aja
	2.01	[2,4)	[-2,0]	[0,2]	bja
2	-	[50,oo)	[0,2)	[14,16)	zkh
	1.06	[40,42)	[0,2)	[14,16)	ukh
	1.09	[50,oo)	[-2,0]	[16,18)	zji

4.5.3 Identification Results

Examples of falsified trajectories and corresponding clustering results are shown in Figure 4.8 (CV penetration rate = 100%). In Figure 4.8(a), a falsified trajectory (marked in red) is generated with the attack goal ETA = 60 s. The falsified vehicle follows the leading vehicle based on Newell’s car-following model. At TOI it slows down to achieve the attack goal. Figure 4.8(b) shows the clustering result. For clarity, the falsified vehicle is always labeled as 1 in all cases. Other vehicles are labeled (starting from 2) based on the time they enter the intersection area. The threshold in the clustering algorithm ϵ is set 6, which is highlighted using a red dashed line. Figure 4.8(b) shows that normal trajectories (trajectory 2 to 8) form one cluster, while the falsified trajectory forms the second cluster. Because of the unusual behavior (slowing down when the front vehicle is still far), the distance between the falsified trajectory and other trajectories is significant. Therefore, the clustering method is able to identify the falsified trajectory. Similarly, in Figure 4.8(c), a falsified vehicle is generated with the attack goal QUEUE = 30 veh. Again, the falsified vehicle follows the leading vehicle based on Newell's car-following model and stops far from the intersection at TOI. This unusual behavior is successfully captured by the proposed method, as shown in Figure 4.8(d). In Figure 4.8(e), a falsified trajectory is generated without an attack goal. The falsified vehicle simply travels at free-flow speed, which appears to be normal. Therefore, based on the predefined threshold ϵ, the falsified trajectory is considered normal. Since this vehicle does not change the values of queue length or ETA, its impact to the CV-TSC system is also minimal.

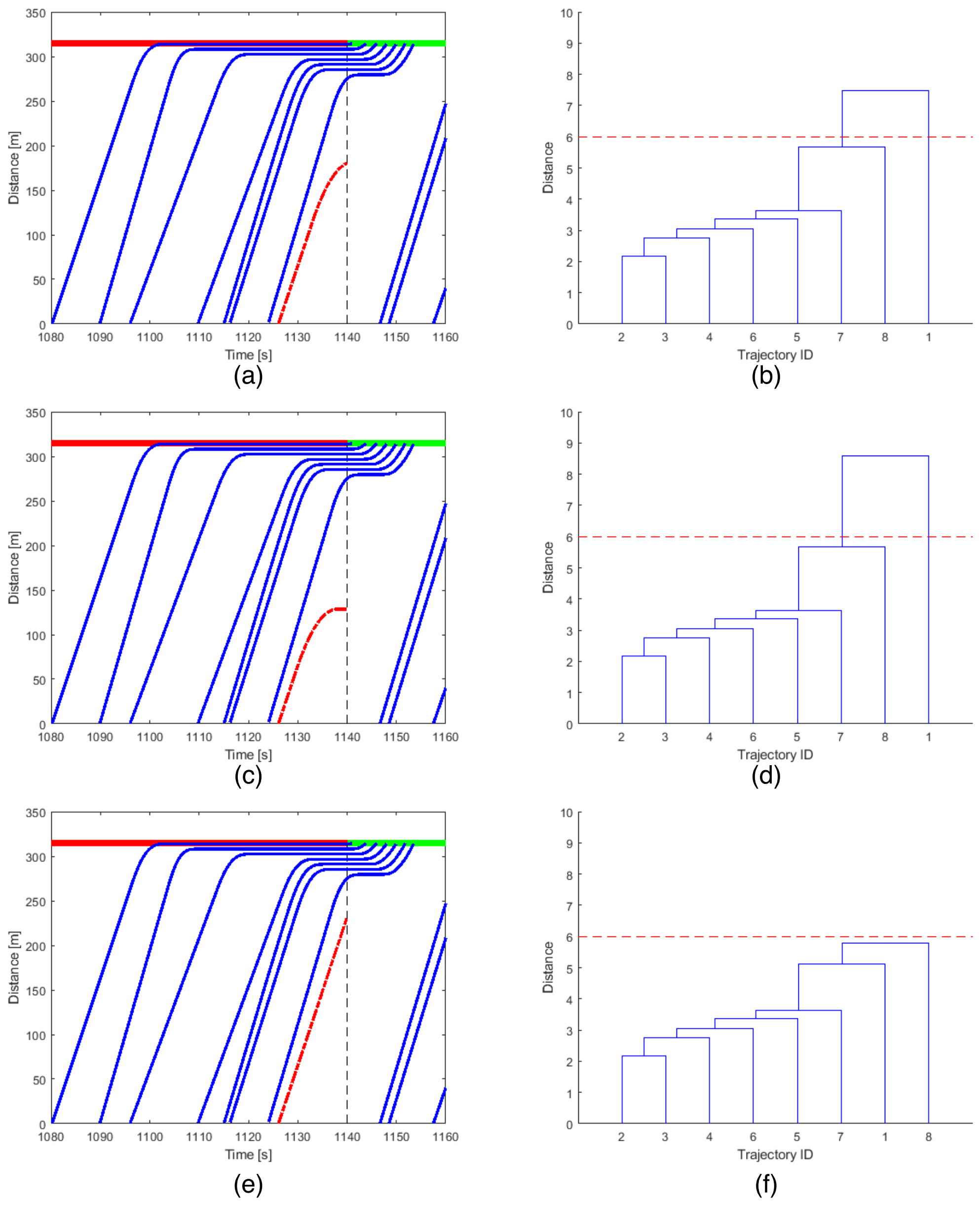


Figure 4.8: Identification results (CV penetration rate = 100%): (a) example of a falsified trajectory (ETA = 60 s); (b) example of hierarchical clustering (ETA = 60 s); (c) (QUEUE = 30 veh); (d) example of hierarchical clustering (QUEUE = 30 veh); (e) example of a falsified trajectory with (without an attack goal); and (f) example of hierarchical clustering (without an attack goal)

Another set of examples are shown in Figure 4.9, where the CV penetration rate is changed to 50%. This means that there are fewer normal trajectories observed every cycle. The proposed method can successfully identify the falsified trajectories even when the penetration rate is low, as shown in Figure 4.9(b) and (d). An interesting finding in Figure 4.9(f) is that even though there is no attack goal when generating the falsified trajectory, the proposed method can still identify the falsified trajectory. This is because the car-following model for generating the falsified trajectory (Newell's car-following model) is different from the car-following model in VISSIM (Wiedemann car-following model). This difference is prominent when the falsified vehicle slows down.

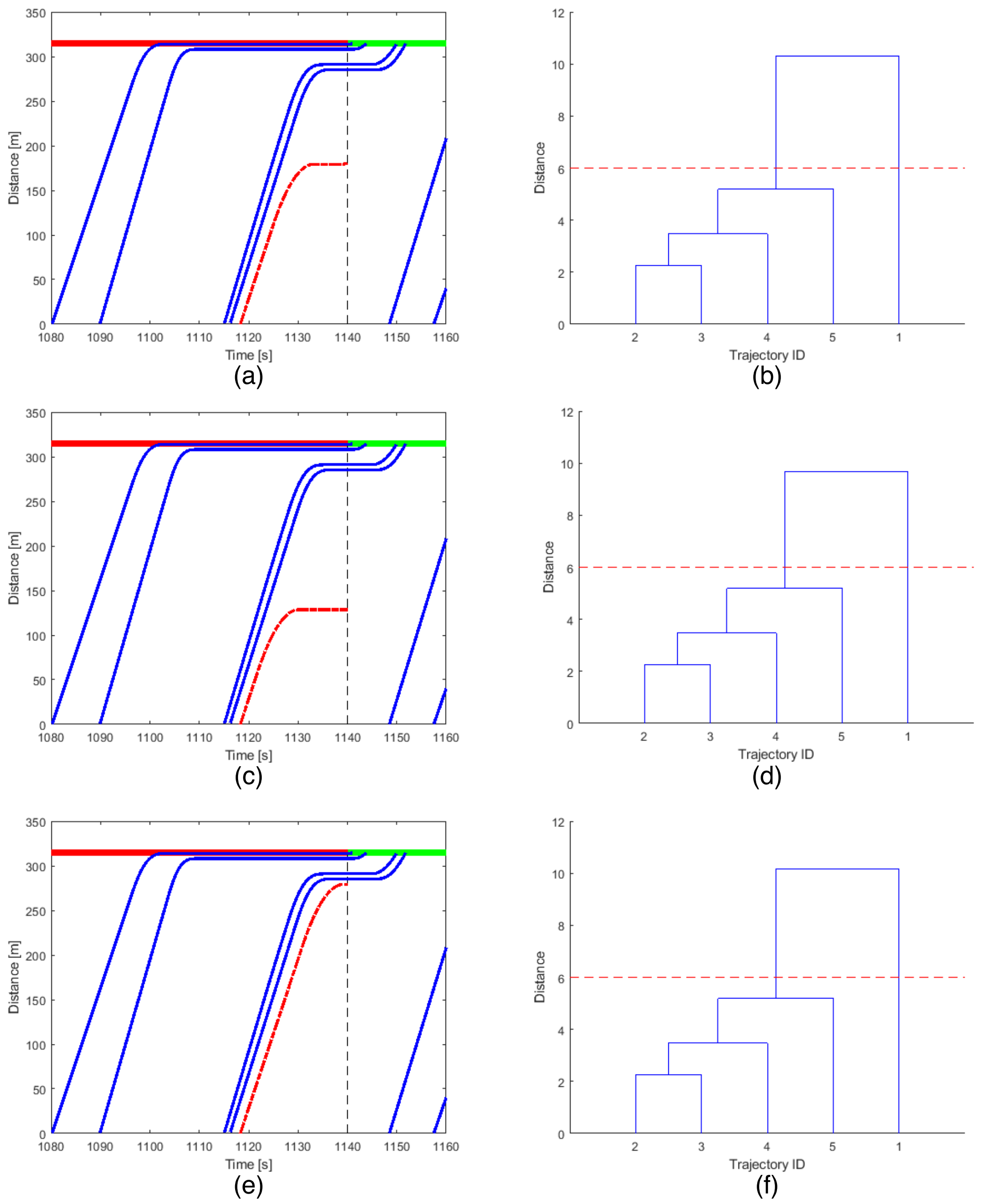


Figure 4.9: Identification results (CV penetration rate = 50%): (a) example of a falsified trajectory (ETA = 60 s); (b) example of hierarchical clustering (ETA = 60 s); (c) (QUEUE = 30 veh); (d) example of hierarchical clustering (QUEUE = 30 veh); (e) example of a falsified trajectory (without an attack goal); and (f) example of hierarchical clustering (without an attack goal)

Table 4.2 shows a summary of the results for falsified trajectory identification. Each case includes 300 signal cycles (i.e., 300 chances to generate falsified trajectories). One falsified trajectory is generated by solving **P2** per signal cycle. However, if there is no feasible solution to **P2**, then no falsified trajectory is generated in this cycle. The attack success rate is defined as the number of generated falsified trajectories divided by the number of total cycles. When the penetration rate decreases from 100% to 50%, the attack success rate increases. This is because less normal trajectories means that the attacker has more space for generating falsified trajectories. When there is an attack goal, generating a falsified trajectory is difficult. This is reflected by the low attack success rate. A valid cycle means a signal cycle in which at least three trajectories (including falsified trajectory) are observed. This is because the clustering algorithm is not meaningful if there are not enough observations. The proposed method is only applied in valid cycles. Detection rate is defined as the ratio of the number of identified falsified trajectories to the total number of falsified trajectories in valid cycles. False alarm rate is defined as the ratio of the number of false identified trajectories (i.e., normal trajectories but labeled as falsified) to the total number of normal trajectories in valid cycles. When there is a specific attack goal, the proposed method can identify most falsified trajectories with a detection rate of at least 99%. However, when there is no attack goal, it becomes difficult to distinguish falsified trajectories from normal trajectories. The attack goal is indeed the reason why the falsified trajectories are abnormal. The proposed method is effective even when the penetration rate is low. The false alarm rates are low for all the cases (less than

7%).

Table 4.2: A summary of results for falsified trajectory identification

PR	Attack goal	Attacks launched	Attack success rate	Valid cycles	% of valid cycles	Detection rate	False alarm rate
100%	ETA = 60s	165	55.0%	300	100.0%	100.0%	2.0%
100%	QUEUE = 30 veh	145	48.3%	300	100.0%	100.0%	1.6%
100%	without a goal	269	89.7%	300	100.0%	14.1%	1.1%
50%	ETA = 60s	214	71.3%	281	93.7%	99.0%	6.6%
50%	QUEUE = 30 veh	200	66.7%	282	94.0%	99.0%	5.9%
50%	without a goal	288	96.0%	288	96.0%	72.4%	6.6%

In the last case (50% PR without an attack goal) the detection rate is still relatively high (72.4%). The major reason is related to the way that the falsified trajectories are generated. In the 100% penetration rate case, the falsified trajectories are generated 2 seconds after the generation of the last CV in the queueing traffic (2 seconds correspond to the saturation flow rate). 2 seconds are chosen because the falsified vehicle will not be too close to its leading vehicle, and at the same time it minimizes the potential risk of conflicting with a following vehicle. For 50% PR case, the same rule (2 seconds) is followed. However, when the penetration rate is low, observing a CV closely following another CV becomes a relatively rare event. That is the reason why it is considered abnormal.

4.5.4 Sensitivity Analysis

A sensitivity analysis is conducted to evaluate the impact of the threshold ϵ on the identification results. The results are shown in Figure 4.10. The first case is considered in the sensitivity analysis (attack goal: ETA = 60 seconds; PR = 100%). A larger threshold indicates more tolerance for abnormal behaviors. Therefore, the proposed method would fail to identify falsified trajectories. When the threshold exceeds 10, the detection rate drops to zero. The false alarm rate also drops to zero. A smaller threshold permits less abnormal behaviors. When the threshold is 6, the detection rate is 100%. The false alarm rate also increases to 2%. As expected, both detection rate and false alarm rate decrease as the threshold increases. This means that there is a trade-off between detection rate and false alarm rate.

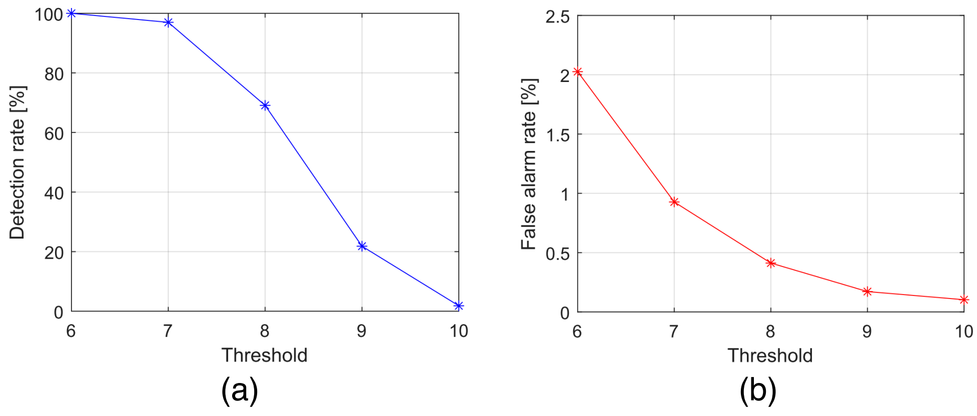


Figure 4.10: The impact of threshold on the results: (a) detection rate; and (b) false alarm rate

4.6 Chapter Summary

To protect CV-TSC systems from falsified data attacks, this chapter proposed a method to identify falsified vehicle trajectories. It was assumed that falsified trajectories need to achieve a certain attack goal. As a result, they were behaviorally distinct from normal trajectories. The problem of identifying falsified trajectories was hence considered an abnormal trajectory identification problem. A trajectory embedding model was developed to generate vector representations of trajectory data points. The similarity (distance) between trajectories was computed based on vector representations. Hierarchical clustering was applied to identify abnormal (i.e., falsified) trajectories.

A series of experiments were then conducted to evaluate the effectiveness of the proposed method. In experiments, falsified trajectories were generated every signal cycle under different penetration rates (i.e., 100% and 50%) and with different attack goals (i.e., ETA attack and phantom queue attack). Findings indicated that the proposed method could successfully identify the majority (over 99% in general) of falsified trajectories, while maintaining a low false alarm rate (under 7% in general).

The proposed defense strategy provides one solution to protecting CV-TSC systems. Other solutions exist as well. One possible solution is to design security-aware algorithms for traffic signal control. For example, Yin (2008) and Zhang et al. (2010) designed robust optimal signal timings that were less sensitive to traffic flow fluctuations. Similarly, engineers and scholars could design robust signal control algorithms that are less vulnerable to falsified data attacks. Another solution is to improve hardware security and prevent attackers from sending falsified CV data. For example, Hu et al. (2020) proposed a system named CVShield, which relocated all related codes from the rich execution environment into the trusted execution environment and ensured the integrity of CV data from reading to transmission. Therefore, attackers could not easily launch falsified data attacks.

Thus far, this dissertation has investigated the cyber security problem from the attacker's perspective and defender's perspective. Falsified trajectories presented in the numerical examples are generated offline because they are only used to validate the proposed defense solution. It is necessary to construct a testing platform that integrates the cyber attack and cyber defense services with an operational CV-TSC system and to conduct attack and defense experiments in real time. These tasks will be explored in the next chapter.

CHAPTER 5

Testing Platform Development

5.1 Introduction

The previous chapters investigated the potential cyber attacks against CV-TSC systems and proposed a defense strategy to protect these systems. Due to the complex nature of real TSC systems and the sensitivity of cyber security research, it would be difficult and unrealistic to implement the proposed attack and defense methods directly at real-world intersections. The cyber security of CV-TSC systems represents a new problem, hence the lack of available platforms (either real-world, simulation, or mixed). Therefore, a testing platform is urgently needed to facilitate the evaluation of different attack and defense models. This chapter leverages the transportation infrastructure in Mcity, University of Michigan's closed testing facility for connected and automated vehicles, to develop such a testing platform. The design objective is to mimic the real traffic environment to the greatest extent, while minimizing the resources required for testing. To achieve this goal, DSRC communication devices in Mcity are utilized to transmit and receive real BSMs and SPaT messages. The traffic environment is modeled by a microscopic simulator VISSIM, in which individual vehicle behaviors can be modeled in detail. The virtual traffic environment and the DSRC communication devices are connected through an augmented reality testing environment (Feng et al., 2018b). Experiments are conducted to compare the system performance with and without attack and defense. I-SIG is the targeted CV-TSC system for experiments. Based on the analysis in Chapter 3, falsified trajectories are generated to influence signal control decisions in attack experiments. In defense experiments, the defense strategy proposed in Chapter 4 is adopted to identify falsified trajectories and protect the CV-TSC system. Moreover, the testing platform is designed generically so different attack, defense, and signal control models can be implemented and evaluated.

The rest of this chapter is organized as follows. Section 5.2 introduces the architecture of the testing platform. Section 5.3 details the experiment setup. Section 5.4 presents the experiment results. Finally, Section 5.5 summarizes this chapter.

5.2 Architecture of the Testing Platform

The overall architecture of the cyber security testing platform is illustrated in Figure 5.1. This testing platform is an extension of the augmented reality testing environment developed by Feng et al. (2018b). The testing platform consists of the following five parts: Traffic Simulator, Mcity, Signal Control System, Attack Program, and Defense Program.

Virtual traffic is generated in the Traffic Simulator. Correspondingly, BSMs and SPaT messages are generated. These messages are forwarded to the infrastructure in Mcity and broadcast by the infrastructure. All real CV communication takes place in Mcity using DSRC technology. The Signal Control System uses received BSMs for traffic signal optimization and controls the traffic signal using NTCIP commands defined by the National Transportation Communications for Intelligent Transportation Systems Protocol (NTCIP, 2020). The Attack Program generates falsified trajectories based on received BSMs and SPaT messages. Falsified trajectories are broadcast in the form of BSMs. The Defense Program is responsible for identifying falsified trajectories.

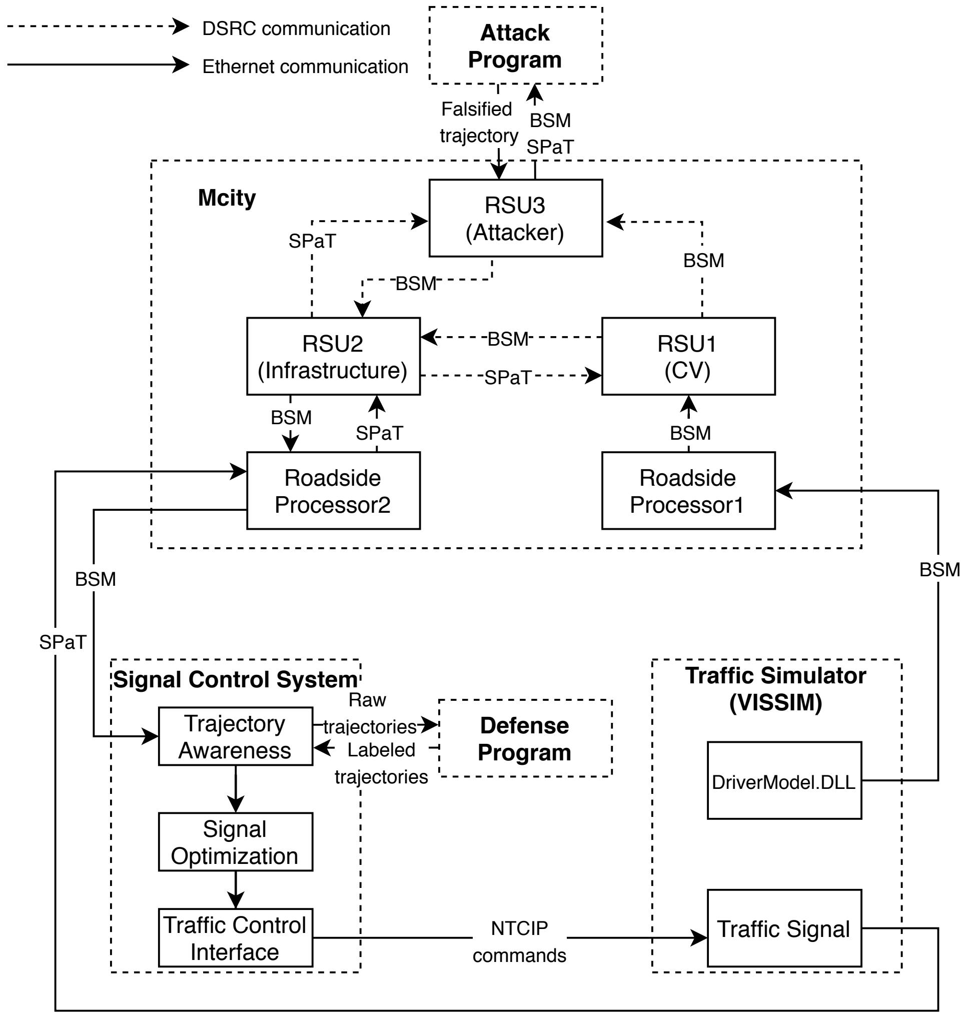


Figure 5.1: Overall architecture of the testing platform

This testing platform uses standard DSRC technology for CV communication, thus maximizing realisticity. Meanwhile, the Traffic Simulator greatly reduces the cost and risk for conducting traffic experiments as there is no real vehicle involved in experiments. The testing platform can be used to test different attack and defense methods, as well as different signal control systems.

5.2.1 Traffic Simulator

VISSIM (Fellendorf and Vortisch, 2010), a commercial software, is used for simulating traffic. A typical 4-leg intersection is modeled (the high-speed intersection located at the freeway segment in Mcity) in VISSIM, as shown in Figure 5.2. Each approach contains one left-turn lane and one through lane. The right-turn movement is not modeled because it is usually not assigned with a signal phase. Each lane is stretched to be 300 meters (the communication range for DSRC). Virtual traffic is generated in this network. Vehicles are generated at the boundary of each approach and move toward the intersection. During the simulation, individual vehicle information such as location and speed is encoded into BSMs every 0.1 second and sent out by DriverModel.DLL.

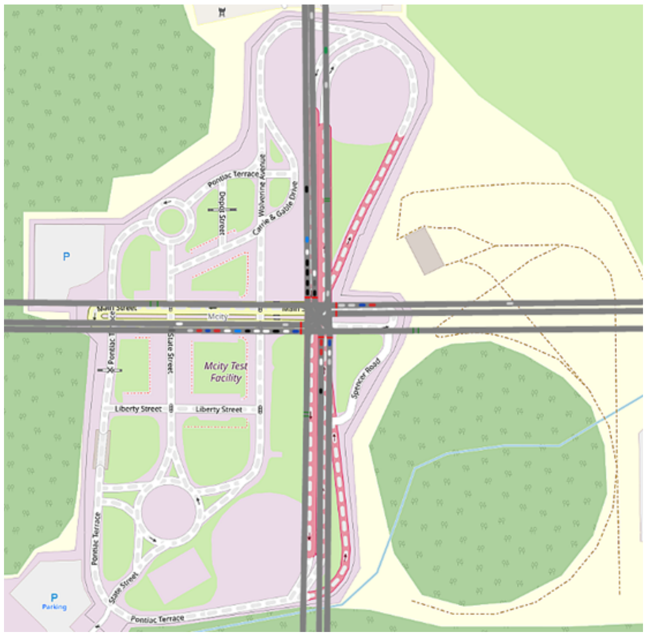


Figure 5.2: Illustration of the signalized intersection modeled in VISSIM

5.2.2 Mcity

Mcity [1] is located on the University of Michigan North Campus in Ann Arbor, Michigan. Mcity occupies 32 acres of land. It is a high-fidelity virtual city that is solely built for testing connected and automated vehicles. It has eight signalized intersections, four of which are equipped with Roadside Units (RSUs) in the downtown area. The RSUs are CV communication devices on the infrastructure side. The communication range of the RSUs covers the whole Mcity area.

The testing platform utilizes Mcity's infrastructure for modeling the DSRC communication network. In Figure 5.1, Roadside Processor1 receives the BSMs from the Traffic Simulator and forwards them to RSU1 via Ethernet. RSU1 plays the role of regular CVs and broadcasts these simulated BSMs via DSRC radio. Meantime, the traffic signal in VISSIM sends SPaT messages to Roadside Processor2, which forwards the messages to RSU2. RSU2 plays the role of transportation infrastructure that connects to the Traffic Control System. It broadcasts SPaT messages via DSRC. The same as in the real-world, CVs (RSU1) can receive SPaT messages from the infrastructure (RSU2) and the infrastructure (RSU2) can receive BSMs from CVs (RSU1). RSU2 sends received BSMs (i.e., vehicle trajectories) to Roadside Processor2, which forwards the messages to the Signal Control System. RSU3 plays the role of an attacker, which receives the same BSMs and SPaT messages that other CVs receive, and forward them to the Attack Program. After the Attack Program generates falsified trajectories, RSU3 broadcasts these trajectories

[1]More information on Mcity can be found at https://mcity.umich.edu/

in the form of BSMs. Note that RSUs are used to replace OBUs in the design. This is because RSUs have the exact technical specifications as the OBUs regarding DSRC communication. The replacement does not bring any impact on the experiment results.

5.2.3 Signal Control System

I-SIG from MMITSS (University of Arizona et al., 2016) is used as the signal control system in the testing platform. I-SIG includes three components: Trajectory Awareness, Signal Optimization, and Traffic Control Interface. The Trajectory Awareness component decodes and temporarily stores the received BSMs (trajectories). Important information such as the requested phase and estimated time of arrival of each vehicle can be extracted from these BSMs. The Signal Optimization component requests trajectory data from the Trajectory Awareness component, computes optimal signal timing plan, and sends it to the Traffic Control Interface component. The Traffic Control Interface component executes the optimal signal timing plan in VISSIM through NTCIP commands.

5.2.4 Attack Program

The Attack Program generates a falsified trajectory for every signal cycle. In Chapter 4, falsified trajectories are generated offline. Different from that, in the Attack Program, real-time falsified trajectories are generated and real-time attacks are launched. To do this, the Attack Program needs to solve the falsified trajectory generation problem in real time based on received BSMs. It is assumed that, by analyzing historical SPaT data, the attacker knows the time that the signal optimization is executed (i.e., TOI t_o). The flowchart of the attack process is illustrated in Figure 5.3. Like regular vehicles, the initial location of the falsified vehicle is at the boundary of an approach and the initial speed is free-flow speed. Starting Δt seconds before t_o, the attacker attempts to solve the falsified trajectory generation problem (i.e., **P2** introduced in Chapter 3.4) every 1 second. The attacker predicts the trajectories of the leading CV and following CV by assuming they keep current speed. If a feasible solution can be found, the attacker initializes the falsified trajectory and starts broadcasting the falsified trajectory in the form of BSMs every 0.1 second. The falsified trajectory is updated by solving **P2** every 1 second. In this way, more recent trajectories of the leading and following CV can be utilized in solving the falsified trajectory generation problem. When solving **P2** for updating, the initial speed and location in **P2** are set to the current speed and location of the falsified vehicle. This ensures that the falsified trajectory is consistent. If no feasible solution can be found, the attacker gives up attacking during this cycle. The attacker stops generating the falsified trajectory after t_o.

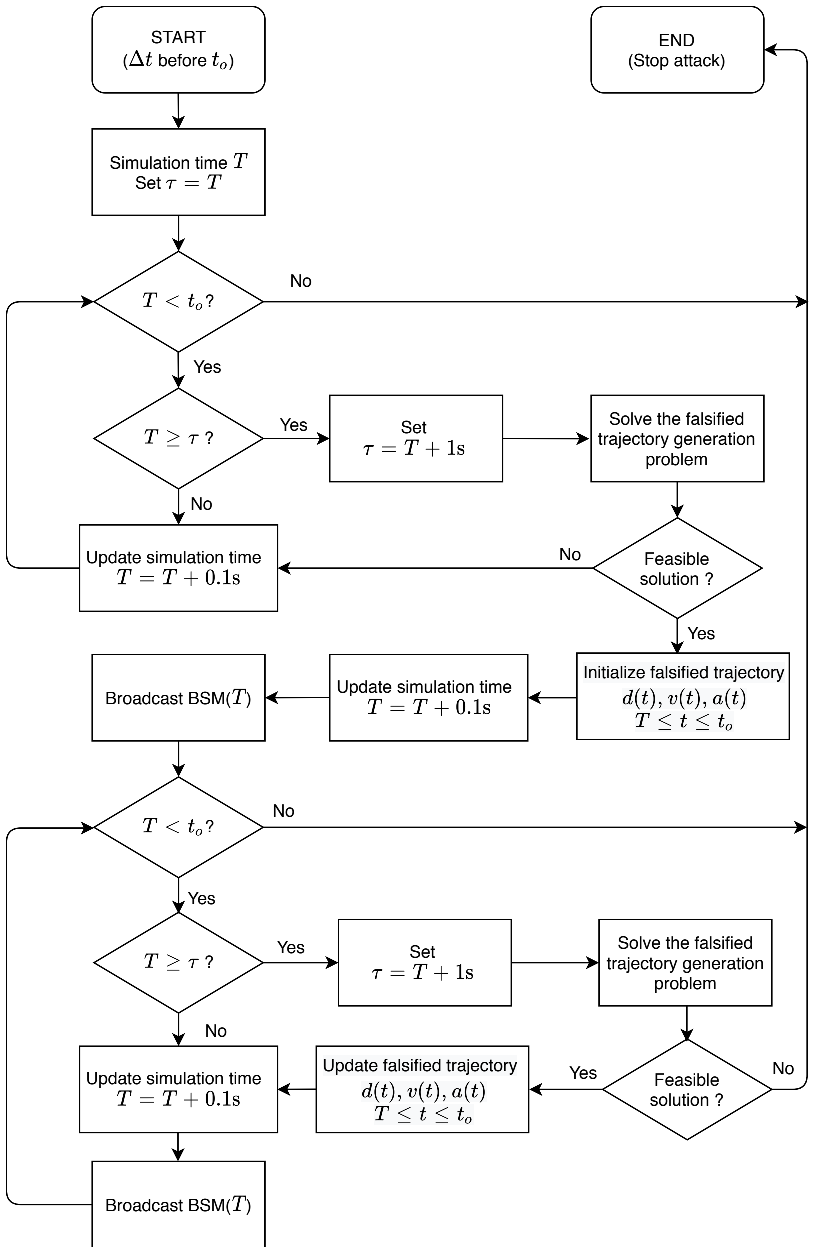

START (Δt before t_o)
END (Stop attack)
Simulation time T Set $\tau = T$
$T < t_o$?
No
Yes
$T \geq \tau$?
Yes
Set $\tau = T + 1$s
Solve the falsified trajectory generation problem
No
Update simulation time $T = T + 0.1$s
No
Feasible solution ?
Yes
Initialize falsified trajectory $d(t), v(t), a(t)$ $T \leq t \leq t_o$
Update simulation time $T = T + 0.1$s
Broadcast BSM(T)
$T < t_o$?
No
Yes
$T \geq \tau$?
Yes
Set $\tau = T + 1$s
Solve the falsified trajectory generation problem
No
Update simulation time $T = T + 0.1$s
Update falsified trajectory $d(t), v(t), a(t)$ $T \leq t \leq t_o$
Yes
Feasible solution ?
No
Broadcast BSM(T)

Figure 5.3: Flowchart of the attack process

5.2.5 Defense Program

Before the raw trajectories are utilized by the Signal Optimization component, the Trajectory Awareness component sends the raw trajectories to the Defense Program. After analysis, the Defense Program sends back labeled trajectories to the Trajectory Awareness component. Only the trajectories labeled real will be utilized by the Signal Optimization component. The flowchart of the defense process is illustrated in Figure 5.4. For each signal phase, the defense program collects trajectories from the Trajectory Awareness component. If there are enough observations (at least 3 trajectories), the defense strategy proposed in Chapter 4 is applied. To reduce the computational load, the defense program computes the similarity between each pair of trajectories based on the last 30 trajectory data points. If there are not enough observations (less than 3 trajectories), the proposed defense strategy is not applied. In this case, penetration rate determines the label of the raw trajectories. When the penetration rate is 100%, the collected trajectories will be labeled falsified and not used for signal optimization. I-SIG will allocate the minimum green time for this phase. Because the traffic demand is low (less than three vehicles) and the allocated minimum green time is sufficient for serving the traffic. When the penetration rate is not 100%, however, the observed trajectories will be labeled true and used for signal optimization. This ensures that the Signal Control System would assign enough green time to serve the traffic of this phase. The drawback is that it also allows the falsified trajectory to extend the green time of this phase.

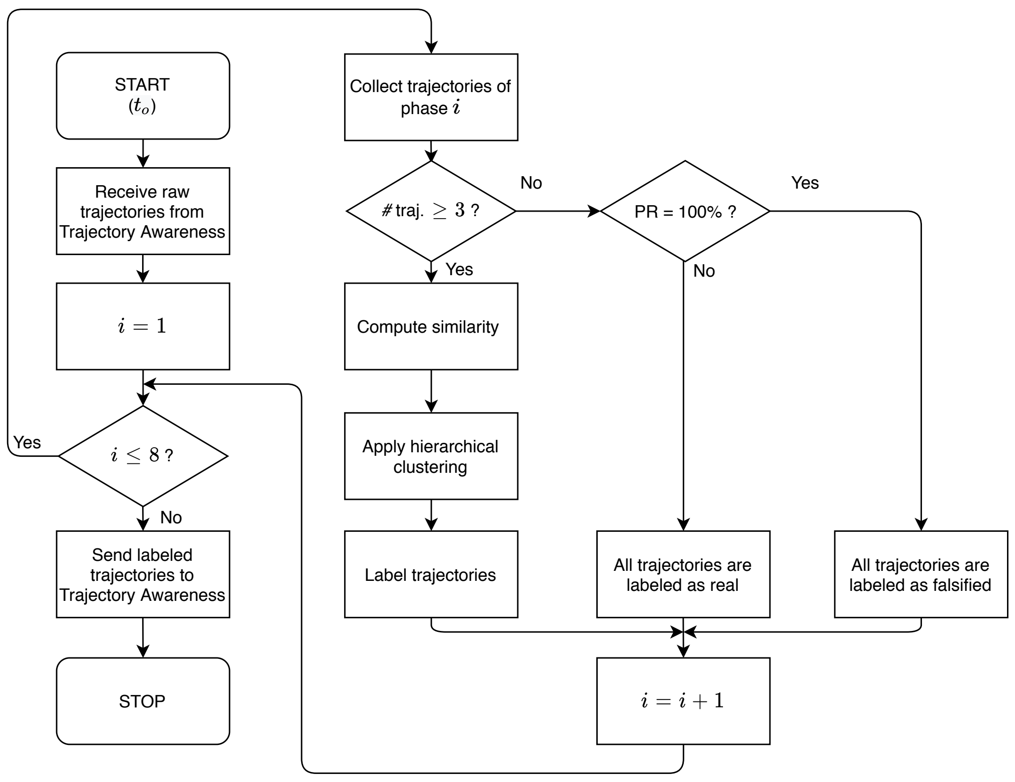

START (t_o)
Receive raw trajectories from Trajectory Awareness
$i = 1$
$i \leq 8$?
Yes
No
Send labeled trajectories to Trajectory Awareness
STOP
Collect trajectories of phase i
#traj. ≥ 3 ?
No
Yes
Compute similarity
Apply hierarchical clustering
Label trajectories
PR = 100% ?
Yes
No
All trajectories are labeled as real
All trajectories are labeled as falsified
$i = i + 1$

Figure 5.4: Flowchart of the defense process

5.3 Experiment Setup

In the Traffic Simulator, the traffic demand for each movement is 350 vehicles per hour. The free-flow speed is 60 km/h. In the Signal Control System, the minimum green time is 5 seconds and maximum green time is 30 seconds for each phase. The yellow time is 2 seconds and all red time is 2 seconds. Phase sequence is fixed (left-turn phases are always executed first). In other words, I-SIG only optimizes phase duration. Each experiment lasts for 1 hour with additional 5 minutes of warm-up time.

Three scenarios are considered. In scenario 1, the CV market penetration rate is set to 100% and ETA attacks are launched. In scenario 2, the CV market penetration rate is set to 50% and ETA attacks are launched. In scenario 3, the CV market penetration rate is set to 50% and phantom queue attacks are launched. For each scenario, three experiments are conducted: normal operation, operation with attack, and operation with attack and defense. It is also assumed that the attacker has limited resources. Consequently, for each signal cycle, one falsified trajectory is generated at Westbound through movement. The parameters for generating falsified trajectories are the same as in Chapter 4. In scenario 1 and 2, the goal of the ETA attack is set to 64 seconds (30 seconds maximum green time for lead phase + 4 seconds transition time + 30 seconds maximum green time for lag phase). In scenario 3, the goal of the phantom queue attack goal is set to 15 vehicles (15 veh $\times$ 2 second/veh saturation flow headway = 30 second maximum green time). In scenario 1 and scenario 2, the parameter Δt is set to 17.5 seconds. In scenario 3, Δt is set to 21.5 seconds. These values ensure that the attacker has enough time to generate falsified trajectories. The threshold ϵ in the hierarchical clustering is set to 8.8.

5.4 Experiment Results

Average vehicle delay is used as the performance index to evaluate the effectiveness of the Attack Program and Defense Program. The experiment results are summarized in Table 5.1. The ETA attack and phantom queue attack are both effective and can downgrade system performance by introducing more delay. The ETA attack is particularly effective, causing 23.0% more delay in scenario 1 and 17.6% in scenario 2. The ETA attack extends lead and lag phases to the maximum. The green time is unnecessarily long for these phases. This causes more delay because all the vehicles in other phases have to wait to be served. The phantom queue attack only extends the attacked phase to the maximum. Therefore, the phantom queue attack is less severe (8.7% delay increase). Experiments 3, 6, and 9 show that the proposed defense strategy can successfully safeguard the system and reduce delay caused by cyber attacks. The delay increase is reduced to 3.8%, 3.3%, and 4.3% respectively.

Table 5.1: Average vehicle delay for each experiment

Scenario	PR	Attack goal	Experiment	Description	Average delay [s/veh]	Delay increase
1	100%	ETA = 64 s	1	Normal operation	39.50	-
			2	Attack w/o defense	48.58	23.0%
			3	Attack w/ defense	41.01	3.8%
2	50%	ETA = 64 s	4	Normal operation	43.38	-
			5	Attack w/o defense	51.03	17.6%
			6	Attack w/ defense	44.80	3.3%
3	50%	Queue = 15 veh	7	Normal operation	43.38	-
			8	Attack w/o defense	47.16	8.7%
			9	Attack w/ defense	45.26	4.3%

To have a better understanding of the performance of the Attack Program and Defense Program, more detailed results for experiments 3, 6, and 9 are presented in Table 5.2. In experiment 3, a total of 42 cycles are recorded and 22 attacks are launched. The attack success rate is 52.4%. 21 falsified trajectories go through the defense component and all of them are successfully identified (1 falsified trajectory is not tested by the Defense Program because less than three trajectories are observed in that phase). The false alarm rate is as low as 1.1%. In experiment 6 and 9, 32 and 30 attacks are launched respectively. The attack success rate increases to 76.2% and 73.2%. This matches expectations because there is more space for generating falsified trajectories in the time-space diagram. 30 and 21 falsified trajectories are tested by the Defense Program. The detection rate is 100% for experiment 6 and 95.2% for experiment 9. The false alarm rate is 4.9% for experiment 6 and 3.7% for experiment 9. When the penetration rate is low, the Defense Program can still identify most falsified trajectories and keep false alarm rate at a low level. The false alarm rate increases because less normal trajectories are observed every signal cycle. The slight delay increase in experiment 3, 6, and 9 is due to two reasons. First, some real trajectories are incorrectly marked falsified, therefore they are not properly served by I-SIG. Second, falsified trajectories are not identified when there are not enough observations. As a result, these falsified trajectories influence signal control decisions.

Another finding from these experiments is that CV market penetration rate is critical to CV-based applications. Generally speaking, CV-based applications work better with a higher penetration rate. This is evidenced by the average delay in experiment 1 (PR=100%) and the average delay in experiment 4 and 7 (PR=50%). What's more, attack success rate is relatively low when the penetration rate is high. This is because the

Table 5.2: Attack and defense results for each experiment

Experiment	Cycles	Attacks launched	Attack success rate	Attack traj. tested by defense	Detection rate	Regular traj tested by defense	False alarm rate
3	42	22	52.4%	21	100.0%	2486	1.1%
6	42	32	76.2%	30	100.0%	1086	4.9%
9	41	30	73.2%	21	95.2%	1071	3.7%

space for generating falsified trajectories shrinks. Finally, with a higher penetration rate, more CV trajectories can be observed every signal cycle. This ensures that the proposed defense strategy can be implemented.

5.5 Chapter Summary

In this chapter, a cyber security testing platform was developed for CV-TSC systems by leveraging the Mcity transportation infrastructure. This testing platform was then used to conduct attack and defense experiments. Based on the analysis in Chapter 3, falsified trajectories were generated in real time to attack I-SIG. The defense strategy proposed in Chapter 4 was applied to filter out falsified trajectories and protect I-SIG. Three scenarios were considered: an ETA attack under a 100% CV penetration rate, an ETA attack under a 50% CV penetration rate, and a phantom queue attack under a 50% CV penetration rate. Findings from a series of experiments showed that cyber attacks against I-SIG could increase average delay by 23.0%, 17.6%, and 8.7% under the three attack scenarios, respectively. The proposed defense strategy could successfully identify and filter out most falsified trajectories. After applying the defense method, the delay increase was reduced to only 3.8%, 3.3%, and 4.3%, respectively. The high detection rate ensured that the majority of the falsified trajectories were excluded from signal optimization. The low false alarm rate was tolerable because it was equivalent to reducing the CV penetration rate by a small percent. These results indicated that the proposed defense strategy could effectively safeguard the CV-TSC system and mitigate potential adverse effects of cyber attacks. The detection rate was high and false alarm rate was low

CHAPTER 6

Conclusions and Future Research

6.1 Research Summary

With the surging development of CVs, vehicle trajectory data are becoming increasingly available. These trajectory data can be used for traffic signal optimization by traffic signal control systems, otherwise known as CV-TSC systems. However, most CV-TSC systems are designed without considering cyber security issues. The connectivity between vehicles and infrastructure (i.e., V2I communications) means that malicious attackers can potentially send falsified data to CV-TSC systems and influence signal control decisions. Such falsified data attacks can result in traffic congestion and have substantial negative effects on intersection operations. This dissertation aimed to systematically understand cyber security problems in CV-TSC systems under falsified data attacks and devise a countermeasure to safeguard these systems. The objectives were accomplished through four studies.

Empirical Study

The empirical study provided insight into TSC system vulnerability and highlighted the need for related cyber security research. Two TSC systems were considered in the study: a conventional actuated TSC system and a CV-TSC system whose control logic was adapted from I-SIG. For the former TSC system, it was assumed that the attacker compromised wireless vehicle detectors to generate fake vehicle calls or cancel real vehicle calls. For the latter TSC system, it was assumed that the attacker could change the number of connected vehicle messages within the RSU's communication range. The attacker's objective was to maximize total system delay under constraints related to budget and attack intensity. To quantify the consequences of these attacks, the cell transmission model was applied to model traffic flow. Numerical examples revealed that both TSC systems were vulnerable to falsified data attacks.

Cyber Attack

The second study investigated how falsified data attacks may be perpetrated. Most studies concerning cyber security problems have considered a white-box attack scenario, in which the attacker is assumed to have full access to the TSC system and/or the control model (e.g., the empirical study in this dissertation). However, this is a strong and unrealistic assumption. The second study in this dissertation therefore focused on a more realistic but challenging black-box attack scenario, in which the control model was unknown to the attacker. This black-box attack consisted of two steps. In the first step, the attacker learned the signal control model using a surrogate model and identified critical traffic features from a list of pre-selected features. The learned surrogate model was then used to predict signal timing plans based on observed real-time critical traffic features. In the second step, the attacker generated falsified trajectories to alter the values of critical traffic features, therefore influencing signal control decisions. The attacker's objective in this case was to maximize the difference between the predicted signal timing plan and the signal timing plan influenced by falsified trajectories. It was assumed that the maximum-difference plan would increase the system delay. The case study indicates that black-box attacks could still bring significant damage to the targeted CV-TSC system even when the control model was unknown to the attacker.

Defense Strategy

Defense is an important aspect of cyber security problems but has often been ignored by the literature. To protect CV-TSC systems from falsified data attacks, the third study proposed a data-driven method to identify falsified trajectories. Falsified trajectories are behaviorally distinct from normal trajectories because they must accomplish a certain attack goal. Thus, the problem of identifying falsified trajectories was considered an abnormal trajectory identification problem in this study. Inspired by a word embedding model, the study developed a trajectory embedding model that created vector representations of trajectory data points. Vector representations enabled computation of the similarity between different trajectories at the intersection level (i.e., a trajectory trace that traveled through an intersection). Hierarchical clustering was applied to identify abnormal trajectories and then group similar trajectories. Abnormal trajectories could ultimately be identified based on a predefined threshold. Numerical examples showed that the proposed method could successfully identify most falsified trajectories under different penetration rates and with different attack goals while maintaining a reasonably low false alarm rate.

Testing Platform

In the final study, a cyber security testing platform for CV-TSC systems was developed. This testing platform was built upon a virtual traffic simulator (VISSIM) and real-world transportation infrastructure in Mcity. The testing platform could be used to evaluate the impact of cyber attacks and the effectiveness of defense strategies. I-SIG was chosen as the targeted CV-TSC system for experiments. Falsified trajectories were generated in real time to influence signal control decisions, and the proposed defense strategy was applied each time I-SIG optimized signal timing plans. Several experiments were performed under different CV penetration rates and different attack goals. Results indicated that falsified data attacks could significantly increase average delay and that the proposed defense strategy could successfully identify and filter out the majority of falsified trajectories, thereby safeguarding the CV-TSC system.

6.2 Future Research

The revelations from this dissertation unveil several avenues for future research.

Real-World Implementation

Econolite has developed the hardware called Connected Vehicle CoProcessor (CVCP) module. The CVCP module provides an interface between the signal controller and

DSRC devices. It enables the implementation of third-party-developed CV applications. The proposed defense strategy can be potentially deployed in the Econolite CVCP module to identify falsified trajectories in real-time.

Attack Joint Control Systems

Recent studies (Yu et al., 2018; Feng et al., 2018a) proposed the joint optimization of traffic signal operation and vehicle trajectory planning within a unified framework. This unified framework extends traffic control from one dimension (either spatial or temporal) to two dimensions (spatial and temporal) and achieves a better overall system performance. This dissertation only focused on the temporal dimension of traffic control (i.e., traffic signals). It would be worthwhile to investigate whether falsified data attacks can influence the behavior of connected and automated vehicles (i.e., the spatial dimension) and whether falsified trajectories can influence the unified control system (i.e., the spatial and temporal dimensions).

Safety

This dissertation mainly considered the mobility aspect of CV-TSC systems. Future studies could investigate safety, another important factor in these systems. For example, cyber attacks have been shown to bring great instability to Cooperative Adaptive Cruise Control platooning (Amoozadeh et al., 2015; Wang et al., 2020) and may cause collisions. Yet it is unclear whether falsified trajectories increase safety risks in CV-TSC systems. This phenomenon warrants additional research.

BIBLIOGRAPHY

AASHTO, ITE, and NEMA (2006). Intelligent transportation system (ITS) standard specification for roadside cabinets. Standard.

Alnasser, A., Sun, H., and Jiang, J. (2019). Cyber security challenges and solutions for V2X communications: A survey. *Computer Networks*, 151:52–67.

Amoozadeh, M., Raghuramu, A., Chuah, C.-N., Ghosal, D., Zhang, H. M., Rowe, J., and Levitt, K. (2015). Security vulnerabilities of connected vehicle streams and their impact on cooperative driving. *IEEE Communications Magazine*, 53(6):126–132.

Beak, B., Head, K. L., and Feng, Y. (2017). Adaptive coordination based on connected vehicle technology. *Transportation Research Record*, 2619(1):1–12.

Bezzina, D. and Sayer, J. (2014). Safety pilot model deployment: Test conductor team report. Technical report.

Bow, S.-T. (2002). *Pattern recognition and image preprocessing*. Marcel Dekker New York.

Brecht, B., Therriault, D., Weimerskirch, A., Whyte, W., Kumar, V., Hehn, T., and Goudy, R. (2018). A security credential management system for V2X communications. *IEEE Transactions on Intelligent Transportation Systems*, 19(12):3850–3871.

Breiman, L. (2017). *Classification and regression trees*. Routledge.

Canepa, E. S. and Claudel, C. G. (2013a). A framework for privacy and security analysis of probe-based traffic information systems. In *Proceedings of the 2nd ACM international conference on High confidence networked systems*, pages 25–32.

Canepa, E. S. and Claudel, C. G. (2013b). Spoofing cyber attack detection in probe-based traffic monitoring systems using mixed integer linear programming. In *2013 International Conference on Computing, Networking and Communications (ICNC)*, pages 327–333. IEEE.

Checkoway, S., McCoy, D., Kantor, B., Anderson, D., Shacham, H., Savage, S., Koscher, K., Czeskis, A., Roesner, F., Kohno, T., et al. (2011). Comprehensive experimental analyses of automotive attack surfaces. In *USENIX Security Symposium*, volume 4. San Francisco.

Chen, C., Zhang, D., Castro, P. S., Li, N., Sun, L., Li, S., and Wang, Z. (2013). iBOAT: Isolation-based online anomalous trajectory detection. *IEEE Transactions on Intelligent Transportation Systems*, 14(2):806–818.

Chen, Q. A., Yin, Y., Feng, Y., Mao, Z. M., and Liu, H. X. (2018). Exposing congestion attack on emerging connected vehicle based traffic signal control. In *Network and Distributed Systems Security (NDSS) Symposium*.

Crash Avoidance Metrics Partners (CAMP) LLC. Vehicle-to-vehicle communications misbehavior detection.

Daganzo, C. F. (1994). The cell transmission model: A dynamic representation of highway traffic consistent with the hydrodynamic theory. *Transportation Research Part B: Methodological*, 28(4):269–287.

Daganzo, C. F. (1995). The cell transmission model, part ii: network traffic. *Transportation Research Part B: Methodological*, 29(2):79–93.

Ernst, J. M. and Michaels, A. J. (2017). Framework for evaluating the severity of cybervulnerability of a traffic cabinet. *Transportation Research Record*, 2619(1):55–63.

Fellendorf, M. and Vortisch, P. (2010). Microscopic traffic flow simulator VISSIM. In *Fundamentals of traffic simulation*, pages 63–93. Springer.

Feng, Y., Head, K. L., Khoshmagham, S., and Zamanipour, M. (2015). A real-time adaptive signal control in a connected vehicle environment. *Transportation Research Part C: Emerging Technologies*, 55:460–473.

Feng, Y., Yu, C., and Liu, H. X. (2018a). Spatiotemporal intersection control in a connected and automated vehicle environment. *Transportation Research Part C: Emerging Technologies*, 89:364–383.

Feng, Y., Yu, C., Xu, S., Liu, H. X., and Peng, H. (2018b). An augmented reality environment for connected and automated vehicle testing and evaluation. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1549–1554. IEEE.

Feng, Y., Zheng, J., and Liu, H. X. (2018c). Real-time detector-free adaptive signal control with low penetration of connected vehicles. *Transportation Research Record*, 2672(18):35–44.

Ganin, A. A., Mersky, A. C., Jin, A. S., Kitsak, M., Keisler, J. M., and Linkov, I. (2019). Resilience in intelligent transportation systems (ITS). *Transportation Research Part C: Emerging Technologies*, 100:318–329.

Ghafouri, A., Abbas, W., Vorobeychik, Y., and Koutsoukos, X. (2016). Vulnerability of fixed-time control of signalized intersections to cyber-tampering. In *2016 Resilience Week (RWS)*, pages 130–135. IEEE.

Ghena, B., Beyer, W., Hillaker, A., Pevarnek, J., and Halderman, J. A. (2014). Green lights forever: Analyzing the security of traffic infrastructure. In *8th {USENIX} Workshop on Offensive Technologies ({WOOT} 14)*.

Goldberg, Y. and Levy, O. (2014). word2vec explained: deriving Mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*.

Goodall, N. J., Smith, B. L., and Park, B. (2013). Traffic signal control with connected vehicles. *Transportation Research Record*, 2381(1):65–72.

Guler, S. I., Menendez, M., and Meier, L. (2014). Using connected vehicle technology to improve the efficiency of intersections. *Transportation Research Part C: Emerging Technologies*, 46:121–131.

He, Q., Head, K. L., and Ding, J. (2012). PAMSCOD: Platoon-based arterial multi-modal signal control with online data. *Transportation Research Part C: Emerging Technologies*, 20(1):164–184.

He, Q., Head, K. L., and Ding, J. (2014). Multi-modal traffic signal control with priority, signal actuation and coordination. *Transportation Research Part C: Emerging Technologies*, 46:65–82.

Hu, S., Chen, Q. A., Joung, J., Carlak, C., Feng, Y., Mao, Z. M., and Liu, H. X. (2020). Cvshield: Guarding sensor data in connected vehicle with trusted execution environment. In *Proceedings of the Second ACM Workshop on Automotive and Aerial Vehicle Security*, pages 1–4.

Jeske, T. (2013). Floating car data from smartphones: What google and waze know about you and how hackers can control traffic. *Proc. of the BlackHat Europe*, pages 1–12.

John, G. H., Kohavi, R., and Pfleger, K. (1994). Irrelevant features and the subset selection problem. In *Machine Learning Proceedings 1994*, pages 121–129. Elsevier.

Kenney, J. B. (2011). Dedicated short-range communications (DSRC) standards in the united states. *Proceedings of the IEEE*, 99(7):1162–1182.

Koonce, P. and Rodegerdts, L. (2008). Traffic signal timing manual. Technical report, United States Federal Highway Administration.

Koscher, K., Czeskis, A., Roesner, F., Patel, S., Kohno, T., Checkoway, S., McCoy, D., Kantor, B., Anderson, D., Shacham, H., et al. (2010). Experimental security analysis of a modern automobile. In *2010 IEEE Symposium on Security and Privacy*, pages 447–462. IEEE.

Laszka, A., Potteiger, B., Vorobeychik, Y., Amin, S., and Koutsoukos, X. (2016). Vulnerability of transportation networks to traffic-signal tampering. In *2016 ACM/IEEE 7th International conference on Cyber-Physical Systems (ICCPS)*, pages 1–10. IEEE.

Lee, J., Park, B., and Yun, I. (2013). Cumulative travel-time responsive real-time intersection control algorithm in the connected vehicle environment. *Journal of Transportation Engineering*, 139(10):1020–1029.

Li, W. and Ban, X. (2018). Connected vehicles based traffic signal timing optimization. *IEEE Transactions on Intelligent Transportation Systems*.

Lighthill, M. J. and Whitham, G. B. (1955). On kinematic waves ii. a theory of traffic flow on long crowded roads. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, 229(1178):317–345.

MATLAB (2018). *9.7.0.1190202 (R2019b)*. The MathWorks Inc., Natick, Massachusetts.

Mazloom, S., Rezaeirad, M., Hunter, A., and McCoy, D. (2016). A security analysis of an in-vehicle infotainment and app platform. In *10th {USENIX} Workshop on Offensive Technologies ({WOOT} 16)*.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Mikolov, T., Yih, W.-t., and Zweig, G. (2013c). Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*, pages 746–751.

Miller, J. R. (2009). Hackers crack into texas road sign, warn of zombies ahead. `https://www.foxnews.com/story/hackers-crack-into-texas-road-sign-warn-of-zombies-ahead`. [Online; Accessed 2020-02-10].

National Electrical Manufacturers Association (2003). NEMA standards publication ts 2-2003. Standard, National Electrical Manufacturers Association.

Newell, G. F. (2002). A simplified car-following theory: a lower order model. *Transportation Research Part B: Methodological*, 36(3):195–205.

NTCIP (2020). The national transportation communications for ITS (intelligent transportation systems) protocol. https://www.ntcip.org/. Online; Accessed: 2020-04-28.

Pandit, K., Ghosal, D., Zhang, H. M., and Chuah, C.-N. (2013). Adaptive traffic signal control with vehicular ad hoc networks. *IEEE Transactions on Vehicular Technology*, 62(4):1459–1471.

Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z. B., and Swami, A. (2017). Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, pages 506–519.

Perrine, K. A., Levin, M. W., Yahia, C. N., Duell, M., and Boyles, S. D. (2019). Implications of traffic signal cybersecurity on potential deliberate traffic disruptions. *Transportation research part A: policy and practice*, 120:58–70.

Priemer, C. and Friedrich, B. (2009). A decentralized adaptive traffic signal control using v2i communication data. In *2009 12th International IEEE Conference on Intelligent Transportation Systems*, pages 1–6. IEEE.

Prigg, M. (2014). Has new york's traffic light system been hacked? researcher claims to be able to control manhattan traffic (and says the same technique will work around the world). https://www.dailymail.co.uk/sciencetech/article-2617228/New-Yorks-traffic-lights-HACKED-technique-work-world.html. [Online; Accessed 2019-03-12].

Řehůřek, R. and Sojka, P. (2010). Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta. ELRA. http://is.muni.cz/publication/884893/en.

Reilly, J., Martin, S., Payer, M., and Bayen, A. M. (2016). Creating complex congestion patterns via multi-objective optimal freeway traffic control with application to cybersecurity. *Transportation Research Part B: Methodological*, 91:366–382.

Richards, P. I. (1956). Shock waves on the highway. *Operations research*, 4(1):42–51.

Rong, X. (2014). word2vec parameter learning explained. *arXiv preprint arXiv:1411.2738*.

SAE International (2016). J2735 dedicated short range communications (DSRC) message set dictionary. Standard, Society of Automotive Engineers.

Sen, S. and Head, K. L. (1997). Controlled optimization of phases at an intersection. *Transportation science*, 31(1):5–17.

Shoukry, Y., Mishra, S., Luo, Z., and Diggavi, S. (2018). Sybil attack resilient traffic networks: A physics-based trust propagation approach. In *Proceedings of the 9th ACM/IEEE International Conference on Cyber-Physical Systems*, pages 43–54. IEEE Press.

Sinai, M. B., Partush, N., Yadid, S., and Yahav, E. (2014). Exploiting social navigation. *arXiv preprint arXiv:1410.0151*.

University of Arizona, University of California PATH Program, Savari Networks, Inc., and Econolite (2016). Multi-modal intelligent traffic signal system-phase ii: System development, deployment and field test.

USDOT (2019a). Connected vehicle pilot deployment program. https://www.its.dot.gov/pilots/. Online; Accessed: 2019-05-21.

USDOT (2019b). Multi-modal intelligent traffic safety system. https://www.its.dot.gov/research_archives/dma/bundle/mmitss_plan.htm. Online; Accessed: 2019-05-21.

Wang, P., Wu, X., and He, X. (2020). Modeling and analyzing cyberattack effects on connected automated vehicular platoons. *Transportation Research Part C: Emerging Technologies*, 115:102625.

Wang, X., Mao, S., and Gong, M. X. (2017). An overview of 3gpp cellular vehicle-to-everything standards. *GetMobile: Mobile Computing and Communications*, 21(3):19–25.

Whyte, W., Weimerskirch, A., Kumar, V., and Hehn, T. (2013). A security credential management system for V2V communications. In *2013 IEEE Vehicular Networking Conference*, pages 1–8. IEEE.

Wong, W., Huang, S., Feng, Y., Chen, Q. A., Mao, Z. M., and Liu, H. X. (2019). Trajectory-based hierarchical defense model to detect cyber-attacks on transportation infrastructure. In *Transportation Research Board 2019 Annual Meeting*.

Wu, J., Ghosal, D., Zhang, M., and Chuah, C.-N. (2017). Delay-based traffic signal control for throughput optimality and fairness at an isolated intersection. *IEEE Transactions on Vehicular Technology*, 67(2):896–909.

Xu, Q.-S. and Liang, Y.-Z. (2001). Monte carlo cross validation. *Chemometrics and Intelligent Laboratory Systems*, 56(1):1–11.

Yang, Z., Feng, Y., Gong, X., Zhao, D., and Sun, J. (2019). Eco-trajectory planning with consideration of queue along congested corridor for hybrid electric vehicles. *Transportation Research Record*, 2673(9):277–286.

Yen, C.-C., Ghosal, D., Zhang, M., Chuah, C.-N., and Chen, H. (2018). Falsified data attack on backpressure-based traffic signal control algorithms. In *2018 IEEE Vehicular Networking Conference (VNC)*, pages 1–8. IEEE.

Yeo, H., Skabardonis, A., Halkias, J., Colyar, J., and Alexiadis, V. (2008). Oversaturated freeway flow algorithm for use in next generation simulation. *Transportation Research Record*, 2088(1):68–79.

Yin, Y. (2008). Robust optimal traffic signal timing. *Transportation Research Part B: Methodological*, 42(10):911–924.

Yu, C., Feng, Y., Liu, H. X., Ma, W., and Yang, X. (2018). Integrated optimization of traffic signals and vehicle trajectories at isolated urban intersections. *Transportation research part B: methodological*, 112:89–112.

Yu, C., Feng, Y., Liu, H. X., Ma, W., and Yang, X. (2019). Corridor level cooperative trajectory optimization with connected and automated vehicles. *Transportation Research Part C: Emerging Technologies*, 105:405–421.

Zeadally, S., Hunt, R., Chen, Y.-S., Irwin, A., and Hassan, A. (2012). Vehicular ad hoc networks (VANETS): status, results, and challenges. *Telecommunication Systems*, 50(4):217–241.

Zhang, D., Li, N., Zhou, Z.-H., Chen, C., Sun, L., and Li, S. (2011). iBAT: detecting anomalous taxi trajectories from gps traces. In *Proceedings of the 13th international conference on Ubiquitous computing*, pages 99–108.

Zhang, L., Yin, Y., and Lou, Y. (2010). Robust signal timing for arterials under day-to-day demand variations. *Transportation Research Record*, 2192(1):156–166.

Zhao, B. and Chen, Q. (2017). Location spoofing in a location-based game: A case study of pokémon go. In *International Cartographic Conference*, pages 21–32. Springer.

Zheng, J., Sun, W., Huang, S., Shen, S., Yu, C., Zhu, J., Liu, B., and Liu, H. X. (2018). Traffic signal optimization using crowdsourced vehicle trajectory data. In *Transportation Research Board 2018 Annual Meeting*.

Zheng, Y. (2015). Trajectory data mining: an overview. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 6(3):1–41.

Zhu, J., Jiang, W., Liu, A., Liu, G., and Zhao, L. (2015). Time-dependent popular routes based trajectory outlier detection. In *International Conference on Web Information Systems Engineering*, pages 16–30. Springer.

Zhu, J., Jiang, W., Liu, A., Liu, G., and Zhao, L. (2017). Effective and efficient trajectory outlier detection based on time-dependent popular route. *World Wide Web*, 20(1):111–134.