UNIVERSITY OF CALIFORNIA,
IRVINE

Security Modeling and Analysis for Intelligent Transportation Systems

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Computer Engineering

by

Anthony Bahadir Lopez

2020

DEDICATION

To my wife Kim, and to my mother and sister, who all believed in me and supported me throughout my PhD journey. And to all those who have played important roles in my academic career, including family, friends, labmates, coworkers, badminton teammates, role models, classmates, professors, and mentors.

TABLE OF CONTENTS

LIST OF FIGURES

Page

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to thank the National Science Foundation for their financial support throughout the five years of my graduate student career. This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. (DGE-1839285) for Anthony Lopez. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

I would also like to make a note of appreciation for the guidance and support from my PhD advisor and committee chair, Professor Mohammad Al Faruque, in helping make this journey a successful one.

Additionally, I thank the PhD committee - Professor Wenlong Jin and Professor Zhou Li - and the rest of the professors who have kindly taken the time to understand my research and provide valuable feedback. And I would like to thank the students Preston Avery Rogers, Matthew Footitt, Lang Qin (Sunny), and Shunsuke Negoro for their assistance in research projects.

Finally, I would like to acknowledge my wife Kim, for always encouraging me to push myself to the limits and be the best that I can be. I would also like to acknowledge my family and my best friends, who have given me the friendship and support I needed to get through challenging moments. Lastly, I would like to thank all those who have stretched out a helping hand to me when I was in need. Having completed this PhD, I aspire to help others who were in a similar position that I was and become a mentor for them.

Acknowledgements to Publishers of Previous Work for Their Permission

©2019 IEEE. Reprinted, with permission, from **A. Lopez**, A. V. Malawade, M. A. Al Faruque, S. Boddupalli and S. Ray, Security of Emergent Automotive Systems: A Tutorial Introduction and Perspectives on Practice, IEEE Design Test, and Dec. 2019.

Lopez, Anthony, Wenlong Jin, and Mohammad Abdullah Al Faruque. "Security analysis for fixed-time traffic control systems." Transportation research part B: methodological (2020): 473-495.

©2020 IEEE. **Lopez, Anthony**, Wenlong Jin, and Mohammad Abdullah Al Faruque. Attack Modeling Methodology and Taxonomy for Intelligent Transportation Systems, IEEE Transactions on Intelligent Transportation Systems, Under Review.

©2017 Springer. Reprinted by permission from Anthony Lopez: Springer, Journal of Hardware and Systems Security, **Lopez, Anthony Bahadir**, et al. "A security perspective on battery systems of the internet of things." Journal of Hardware and Systems Security (2017): 188-199.

VITA

Anthony Bahadir Lopez

EDUCATION

Doctor of Philosophy in Computer Engineering **2020**
Masters of Science in Computer Engineering **2017**

University of California Irvine *Irvine, California*

Bachelors of Science in Computer Engineering **2015**
University of California San Diego *San Diego, California*

RESEARCH EXPERIENCE

Graduate Research Assistant **2015–2020**
University of California, Irvine *Irvine, California*

Undergraduate Research Assistant **2013–2015**
University of California, Irvine *Irvine, California*

REFEREED JOURNAL PUBLICATIONS

TileChain: A Geographic Blockchain for V2X Communications **2020**
Journal of IEEE Transactions on Intelligent Transportation Systems (*Under Review*)

Attack Modeling Methodology and Taxonomy for Intelligent Transportation Systems **2020**
Journal of IEEE Transactions on Intelligent Transportation Systems (*Under Review*)

Security analysis for fixed-time traffic control systems **2020**
Transportation Research Part B: Methodological

Gan-sec: Generative adversarial network modeling for the security analysis of cyber-physical production systems **2019**
Design, Automation & Test in Europe Conference & Exhibition (DATE)

A security perspective on battery systems of the internet of things **2017**
Journal of Hardware and Systems Security

REFEREED CONFERENCE PUBLICATIONS

Eve, You Shall Not Get Access! A Cyber-Physical Blockchain Architecture for Electronic Toll Collection Security **2020**
IEEE Intelligent Transportation Systems Conference (IEEE ITSC 2020)

Polarization mode dispersion-based physical layer key generation for optical fiber link security **2017**
Optical Sensors, Optical Society of America

A physical layer security key generation technique for inter-vehicular visible light communication **2017**
Signal Processing in Photonic Communications, Optical Society of America

Exploiting wireless channel randomness to generate keys for automotive cyber-physical system security **2016**
ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS)

OTHERS

Poster: Physical Layer Key Generation Protocol for Secure V2X Communication Architecture **2019**
Network and Distributed System Security Symposium (NDSS), Internet Society

Master Thesis: Physical layer key generation for wireless communication security in automotive cyber-physical systems **2017**
UC Irvine

Poster: Physical Layer Key Generation Protocol for Secure V2X Communication Architecture **2017**
Network and Distributed System Security Symposium (NDSS), Internet Society

PATENTS

Systems and methods for encrypting communication over a fiber optic line **2020**
US Patent App. 16/509,999

Systems and methods for encrypting communication between vehicles **2020**
US Patent App. 16/510,097

Secret key for wireless communication in cyber-physical automotive systems **2018**
US Patent 10,129,022

SOFTWARE

ITS Security Analysis Tool `https://github.com/AICPS/LQM_traffic_sec_official`
Matlab and Python code that permit the study of various attack models on connected fixed-time traffic control systems.

ITS Attack Modeling Taxonomy `https://github.com/AICPS/ITSAttackModeling`
Matlab and C++ code (built upon Veins simulation tool) that permit the study of various attack models on a connected ITS use-case: V2X Advisory Speed Limit Control.

ABSTRACT OF THE DISSERTATION

Security Modeling and Analysis for Intelligent Transportation Systems

By

Anthony Bahadir Lopez

Doctor of Philosophy in Computer Engineering

University of California, Irvine, 2020

Professor Mohammad Al Faruque, Chair

Massive deployment of embedded systems including various sensors, on-board and road-side computing units, wireless communication among vehicles and infrastructure via enabling technology of the Internet of Things (IoT), and intelligent algorithms are changing the transportation sector, leading to novel systems known as Intelligent Transportation Systems (ITS). However, with these newer technologies come unforeseen safety and security concerns. Research has shown that effects of attacks on the peripherals or the Electronic Controller Units (ECUs) in modern vehicles may cause congestion, but more importantly endanger passengers and passersby. An attack vector on one component/subsystem of an ITS will tend to lead to effects on other components/subsystems due to the connectivity between them. In 2014, a whopping 15000+ existing wireless devices (sensors and controller) deployed across 45 U.S. states and 10 countries were found to be potentially vulnerable to exploits that could lead to remote modifications of traffic timing control. The situation will be exacerbated in the future with the increase in autonomy level and the reliance on sensors and communications.

In spite of the recent interest and importance of ITS security, there have been few efforts to consolidate, structure, and unify this large body of research. There has also been an increasing divergence between academic research and industrial practice in the area, each of which has evolved independently with little interaction and in some cases with little understanding of the assumptions, issues, trade-offs, and scales considered by the other. In addition to a lack of a clear consolidation and summary of related ITS security works, research on modeling/analysis tools for ITS security is also lacking.

For these reasons, this dissertation tackles these challenges by providing 1) a consolidation in ITS security research in terms of both V2X and IoT aspects (with a focus on battery systems) and 2) two methodologies to model and analyze the performance of ITS under attacks. Both methodologies are designed to be standalone open-sourced tools that ITS designers, engineers, and researchers may utilize to promote the growth of ITS security The first methodology focuses on modeling attacks and analyzing their impacts on vulnerable connected Fixed-Time Traffic Signal Control Systems. The second methodology is presented hand-in-hand with an attack taxonomy that focuses on a more advanced ITS system use-case called Vehicular Communication (V2X) Advisory Speed Limit Control (ASL) and involves the study of various attack types on different components of the ITS.

Chapter 1

Introduction

1.1 Background

Transportation systems play a major role for goods, services, and people. To ensure that transportation systems are fulfilling their role to the fullest, an extensive amount of work has been put into traffic management schemes to reduce or prevent congestion [206, 200]. Congestion arises when the demand for a certain part of the transportation infrastructure is greater than the services/supply it may provide. Preventing or reducing traffic congestion may reduce environmental and health issues, improve safety, and help save money and time for people and services. In fact, according to a 2010 White Paper by the European Commission [92], the "external costs of road traffic congestion alone amount to 0.5% of Community GDP [Europe]" and will continue to increase if nothing is done to mitigate its impacts [206].

The last few decades have seen a transformation in both automotive and transportation systems from mechanical or electro-mechanical systems to electronic, software-based systems.

According to the U.S. Department of Transportation Federal Highway Administration, "on average, adaptive signal control technologies improve travel time by more than 10%. In areas with particularly outdated signal timing, improvements can be 50% or more" [201]. Furthermore, the U.S. Department of Transportation has established a long-term Intelligent Transportation Systems (ITS) program to encourage the widespread use of ITS across the nation [200]. ITS are transportation-centric Cyber-Physical System (CPS) that are combinations of subsystems that work with one another to improve transportation performance. Each subsystem is made up of loops between sensors of physical phenomena (e.g., number of vehicles, vehicle speeds, vehicle presence), controllers and traffic agencies. To fully grasp a comprehensive understanding of the performance of ITS, each of its subsystems requires individual modeling and analysis.

To aid existing and traditional traffic control systems and to prepare for upcoming implementations of ITS, the Internet of Things (IoT) has become an enabling technology. Wireless communication is being used as for the remote control over the signal timings of one or several intersections, and the transfer of information between sensors, controllers, and traffic management agency servers [74]. Although there has been a shift toward implementing ITS, fixed-timing plans are still the most common throughout the world due to legacy and regulatory issues. In 2008, 90% of traffic control systems throughout the U.S. were fixed-timing control systems [157].

On the other hand, modern automotive systems are complex distributed systems involving the coordination of hundreds of Electronic Control Units (ECUs) communicating through a variety of in-vehicle networks and the execution of several hundred Megabytes of software.

First, the systems are cyber-physical: the ECUs coordinate, monitor, and control a variety of sensors and actuators including LIDAR, cameras, radar, light matrices, devices for sensing angular momentum of the wheels, devices for automated brake and steering control, etc. Second, many computation and communication tasks across the different ECUs, sensors, and actuators must be accomplished under hard real-time requirements; e.g., a pedestrian detection algorithm must complete a slew of complex activities including the capture of sensory data, aggregation, communication, analytics, image processing, security analysis, etc., within the time constraints to enable successful completion of the appropriate actuarial response such as warning the driver or automatically braking.

Furthermore, the complexity is anticipated to rise sharply with increasing autonomy levels in vehicles. For instance, a future self-driving car with autonomy level 4 will include several elements not available in today's (level 2) systems. Example elements include: 1) vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communications with a variety of networks of different levels of trustworthiness, 2) a diversity of sensors to detect driving conditions (e.g., potholes, moisture, pedestrians, etc.), and 3) distributed computing elements to perform in-vehicle analytics and react to evolving conditions on the fly.

Security is therefore obviously of paramount importance for automotive systems. Given that the system involves the complex interaction of sensory, actuarial, and computational elements, an "innocent" misconfiguration or error in one component may result in a subtle vulnerability that can be exploited in-field with potentially catastrophic consequences. Recent work has shown that it is viable, and even relatively straightforward, to hack a vehicle remotely, get control over its driving functionality, and cause an accident. The situation will be exacerbated in the future with the increase in autonomy level and the reliance on sensors and communications: an attacker may hack a vehicle remotely through the interception or tampering of sensor data and/or V2V and V2I messages without requiring physical access or even proximity to the vehicle under attack, thus resulting in a sharp increase in the attack surface.

Consequently, the proliferation and adoption of connected, autonomous, self-driving cars critically depends on our ability to ensure that they perform securely in a potentially adversarial environment. Unsurprisingly, there has been a large interest in recent years in the security of automotive systems, with a flurry of publications demonstrating a diversity of security vulnerabilities and exploits, as well as techniques for defense against these vulnerabilities.

1.2 Research Challenges

Unfortunately, in spite of the interest and importance of ITS security, there has been few efforts to consolidate, structure, and unify this large body of research. Consequently, publications in the area typically appear isolated approaches for specific attacks or defenses, rather than a disciplined study of security challenges or systematic approaches to counter them. Furthermore, much of the research on automotive security is conflated with other related areas on security assurance with analogous but different challenges, including wearables, the Internet of Things, or even traditional hardware and software designs.

Finally, there has been an increasing divergence between academic research and industrial practice in the area, each of which has evolved independently with little interaction and in some cases with little understanding of the assumptions, issues, trade-offs, and scales considered by the other. All this leaves a researcher getting initiated in this area with the daunting tasks of sifting through the various challenges, complexities, and research directions; identifying approaches applicable to automotive systems in particular; and comprehending evolving challenges caused by the rising complexity of these systems through the past, present, and future.

In addition to a lack of a clear consolidation and summary of related ITS security works, research on modeling/analysis tools for ITS security is also lacking. For this reason, two new methodologies to model and analyze the performance of ITS under attacks are presented in this dissertation. The first methodology focuses on modeling attacks and analyzing their impacts on Fixed-Time Traffic Signal Control Systems (the most popular type), which are exposed to security vulnerabilities due to added connectivity. The second methodology focuses on a more advanced ITS system use-case called Vehicular Communication (V2X) Advisory Speed Limit Control and involves the study of various attack types on different components of the ITS. Both methodologies are designed to be standalone open-sourced tools that ITS designers, engineers, and researchers may utilize to promote the growth of ITS security.

1.3 Overview of Contributions

As aforementioned, this dissertation will tackle the existing research challenges of: 1) providing a better understanding of the security of the ever-growing ITS in terms of both automotive and IoT security, 2) developing unique tools to assist ITS designers and engineers to better analyze and develop security solutions, a.The structure of this dissertation will be as follows:

- *Chapter 2* will provide an extensive overview and taxonomy of ITS security challenges and related works.
- *Chapter 3* will provide a detailed methodology on modeling and analyzing attack impacts on Connected Fixed-Time Traffic Signal Control Systems.
- *Chapter 4* will similarly provide a methodology for the ITS use-case: V2X Advisory Speed Limit Control.
- *Chapter 5* will discuss the cyber-physical risks involved with battery system security in ITS.
- *Chapter 6* will conclude the dissertation with some remarks on the contributions and on ideas for future research.

Chapter 2

A Comprehensive Overview and Taxonomy of ITS Security

2.1 Introduction

This chapter represents a first step to provide a comprehensive, systematic overview of both research and practice in ITS security, with a focus on automotive and V2X security. A systematic categorization of research advances in various aspects of both attacks and defenses on automotive electronics is developed and presented. Furthermore, this chapter discusses current practices in security assurance, points out their constraints and trade-offs, and provides perspective on the rationale involved. In short, this chapter serves as a research contribution for an ITS researcher, designer, or engineer to begin investigation on all aspects of the security of connected and autonomous vehicles.

The remainder of the chapter is organized as follows. Section 3.2.1 provides a brief overview of automotive electronics and software to provide a sense of the complexity, potential vulnerabilities, and attacker entry points. Different components of electronic and software system security are also recounted, to provide a background for the rest of the chapter. In Section 2.3, some challenges in mitigating security vulnerabilities in automotive systems are presented. Sections 2.4, 2.5, 2.6, and 2.7 describe various categories of research in automotive security. In Section 2.8 one specific, celebrated vehicle hack is closely examined, e.g., by Miller and Valasek in 2015; the goal of this case study is to examine how vulnerabilities at different levels can be "chained" together by a hacker to compromise a vehicle. Section 2.9 consists of a high-level overview of industrial practice in identifying security vulnerabilities. Section 2.10 briefly discusses security challenges with the automotive supply chain. Section 2.11 concludes the chapter. Note that while this chapter comprehensively discusses automotive security at the vehicle level, automotive security involves many other components, e.g., validation of individual SoCs and ECUs in automobiles, the relation between security and functional safety, etc. These components are outside the scope of this chapter and previously published literature [219, 217, 215, 216] for these topics are referred.

2.2 Background

2.2.1 Electronics and Software in Modern Automotive Systems

The transformation of automotive systems from a mechanical or electro-mechanical system to a chiefly electronic one arguably began with the development of engine control and fuel injection systems in the 1970s. Starting from the 1990s, the design complexity of automotive systems has been dominated by electronic parts, with more focus on software components in the last decade. Today's cars include electronics and software for infotainment, driver assistance (ADAS), and energy efficiency (e.g., emission control), to name a few. The electronic and software components in an automobile (which is loosely referred to as "electronics") are typically divided into five functional domains:

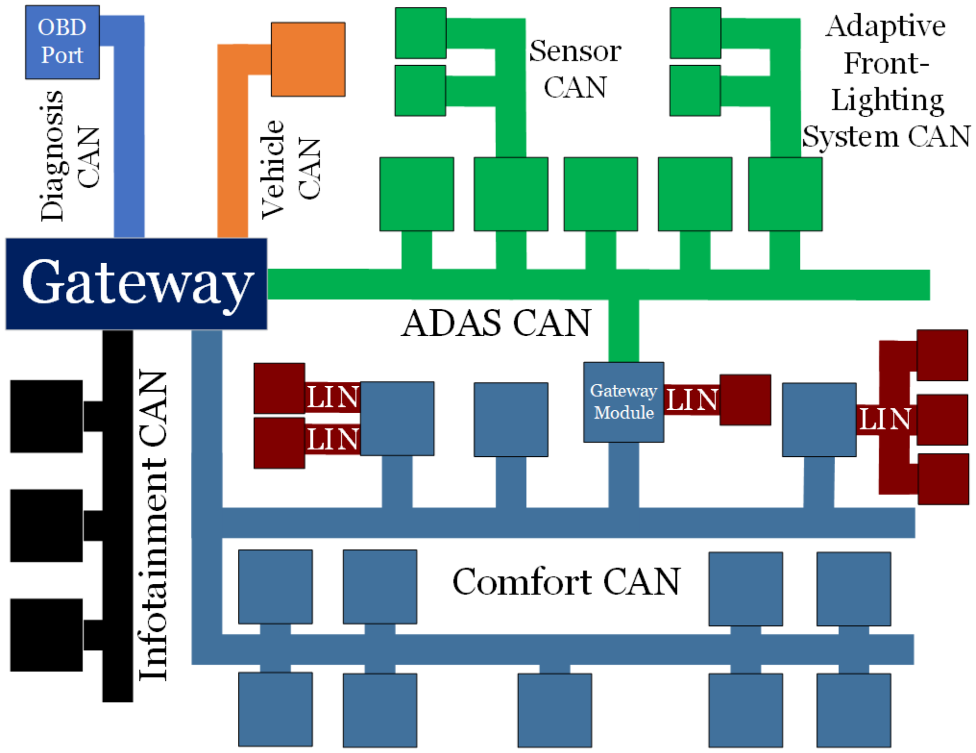


Figure 2.1: Overview of an automotive system architecture. Each box refers to an ECU (controller). ECUs are connected with one another through buses and intra-networking protocols such as Controller Area Network (CAN) and Local Interconnect Network (LIN). CAN is primarily used for core driving functionality and engine control as well as for sensors, comfort, infotainment, and the Adaptive Front-Lighting System.

- **Telematics:** This includes the multimedia and infotainment components of the car including radio, rear-seat entertainment, and navigation systems.
- **Body:** This includes air-conditioning and climate control, the electronic dashboard, power doors, seats, windows, mirrors, lights, park distance control, etc.
- **Chassis:** This includes features such as the Antilock Braking System, Stability Control, Adaptive Cruise Control, etc.
- **Powertrain:** This includes the electronics for controlling the engine, fuel injection, transmission gear, ignition timing, etc.
- **Passive Safety:** This includes all the electronics designed to add safety mechanisms, including roll-over sensors, airbags, belt pre-tensioners, etc.

Obviously, many automotive features cross-cut a variety of functional domains. For example, many modern cars include speed-compensated volume adjustment, i.e., adjustment of multimedia volume in response to increasing speed of the car. This requires communication between the radio (part of telematics) and ADAS components. Other similar examples include automatic braking while reversing if the backup camera senses a child or small obstacle and showing the reversing trajectory on display (which requires computation of angular momentum of the wheels). To enable these features, automotive system architectures involve significant and complex communication among the different in-vehicle components. This is implemented through a variety of protocols including Controller Area Network Bus (CAN-Bus), Local Interconnect Network (LIN), FlexRay, and Media Oriented Systems Transport (MOST). Figure 2.1 provides a representative automotive architecture. In addition to in-vehicle communication, current and emergent vehicles also communicate with external entities (e.g., other cars, infrastructure components, etc.).

2.2.2 Security Requirements

Traditionally, the security of a Functional Safety of Electrical/Electronic/Programmable Electronic Safety-related (E/E/PE) System includes the following foundational pillars: confidentiality, integrity, and availability, also referred to as the *CIA pillars* [101]. More recently, authentication and repudiation have been added as additional pillars, particularly for communicating systems and devices.

1. **Confidentiality:** This refers to the requirement that sensitive, critical system information and data are not perceivable by parties who are not the intended recipients.
2. **Integrity:** This refers to the requirement that an unauthorized entity cannot corrupt or modify sensitive data or information. In the context of communicating agents, integrity involves the requirement that data received is not different than what was originally intended to be sent. Further, the data should be accompanied by a warranty that it was sent from the expected user at an expected time.
3. **Availability:** This refers to the requirement that a legitimate user or application can access requested resources and perform functions within a guaranteed time limit. An obvious subversion on availability is a denial-of-service (DOS) attack.
4. **Authentication:** The assurance that communicating parties can verify the identity of each other and that parties are only able to attain access to resources corresponding to their access level.
5. **Non-Repudiation:** This refers to the assurance that a party cannot refute something they have done (e.g., sending a packet). It requires a mechanism to prove the history of a communicating party. This usually involves a combination of authentication and integrity.

Of course, the above requirements are very general. Translating them for a specific application involves definitions of *security policies* targeted towards that system, e.g., confidentiality requirements are enforced through security policies that stipulate how sensitive assets in the system can be accessed and the agents and devices authorized to access them [28, 29, 219]. Nevertheless, the five pillars above can be used to systematize and categorize security attacks and defenses. For this chapter, when discussing security vulnerabilities on automotive systems, this taxonomy will be used to categorize both attacks and defenses.

2.3 Some Challenges with Automotive Security

At a high level, security attacks on automotive systems are obvious instances of general cybersecurity problems. In particular, a large number of electronic and software components that were not originally designed to be connected to the Internet are now connecting to the Internet, so it is unsurprising that security vulnerabilities exist which can then be exploited in the field. On the other hand, one challenge is that traditional cybersecurity solutions cannot be directly used to mitigate such attacks. Just has been shown in other CPS in research and real-life incidents regarding manufacturing, nuclear, sewage, train, bio-engineering, and others [59, 61, 60, 58, 170, 146, 246, 10, 81].

In particular, automotive systems are in the field for a long time compared to traditional information technology (IT) systems, mobile systems, and wearable devices. For instance, a mobile phone remains in the field for a couple of years. In contrast, a car may remain in the field for a decade or more. This gives the hacker a long time to find vulnerabilities in deployed vehicles. Furthermore, even if there is no security problem at the time of deployment, security requirements can change within this long life-time, adversely impacting the level of security assurance on a deployed, mature system. Furthermore, traditional security assurance solutions, e.g., encryption, authentication, etc., are typically computationally intensive. It is

First Author	Year	Experimental Surface	Citation
Koscher	2010	Unknown	[158]
Checkoway	2011	Unknown	[51]
Foster	2015	Mobile Devices Ingenierie TCU used by Uber	[85]
Miller	2012	2010 Ford Escape & 2010 Toyota Prius	[189]
Miller	2015	2014 Jeep Cherokee	[190]
Keen Security Lab	2016-19	Tesla Model S, BMW	[5], [6], [7], [8]
Smith	2016	2006 Chevrolet Malibu	[247]

Table 2.1: Summary of works that have hacked on-road vehicles to study their vulnerabilities and potential exploits

difficult to deploy many of these solutions with the memory and computation constraints of automotive ECU's. Finally, such solutions may raise issues related to privacy. For instance, in connected platoons, strong authentication may be desirable to ensure that a vehicle-to-vehicle communication is indeed coming from an authentic vehicle; however, a strong authentication scheme may disclose the identity of the sending vehicle, which can then be used to extract various private information including location and driving history.

2.4 A Sampling of Automotive Security Attacks

For the last decade, researchers and white-hat hackers have been experimenting on advanced automotive systems to analyze them and discover vulnerabilities. Their primary purpose in doing this is to showcase the need for secure automotive systems as more and more capabilities are added to them over time. Each entity has used unique methods and experimental setups to conduct their studies. One of these hacks will be studied in-depth in Section 2.8. In this section, a high-level overview of the different hacks is provided to give a general flavor of the spectrum of techniques involved. Table 2.1 provides a summary of these hacks and the related publications organized by author, year, experimental surface (vehicle type) and citations.

One of the earliest comprehensive attacks on an in-field automobile (they did not publish details of the vehicle involved) was performed by Koscher *et al.* [158] in 2010. The work primarily involved exploits based on physical attacks. Subsequently, two other chapters were published following up on the original hack, e.g., by Checkoway *et al.* [51] and Foster *et al.* [85]. Checkoway *et al.* expanded upon Koscher *et al.*'s work with remote exploits that take advantage of the vehicle's telematics system. Foster *et al.* provide a thorough vulnerability analysis of the in-vehicle systems. They considered vulnerabilities of newly introduced (at the time of publication of their work) technologies including new telematic control units (TCU) to both direct and remote attacks.

A landmark study in automotive hacking was performed in 2015 by Miller and Valasek [190] who exhibited a way to remotely control the driver assistance system of a 2014 Jeep Cherokee and drive it off the road. This work is particularly relevant to the research community since it provides detailed documentation and explanation to enable the reproduction of their results.

In addition to the above, there has been work by white-hat hackers and research teams to discover and perform exploits on automotive systems primarily to facilitate research and awareness. One such team: the Keen Security Lab of Tencent [9], discovered exploits on several models of the Tesla Model S, including remote attacks through the CAN module and firmware over-the-air (OTA) updates. They also performed a thorough assessment of in-vehicle equipment in several BMW vehicle models and found many vulnerabilities. Another organization known as the Car Hacking Village [167], comprising several DefCon hackers, published a detailed guide for the ethical hacking of automotive systems [247]. They also include a list of recommended equipment to use. They demonstrate their approach on a 2006 Chevrolet Malibu Sedan, although their techniques apply to other vehicles.

2.5 Security of In-vehicle Networks

The functionalities of automotive systems are typically implemented through the communication and coordination of ECUs and MCUs across several in-vehicle networks. For obvious reasons, these in-vehicle networks are the primary targets of automotive security exploitation: the goal is typically to "fool" the networks into communicating or delivering unauthorized messages. Since many of the messages carried by these networks can have significant impacts on vehicular functionality, e.g., messages through the CAN network can affect vital driving functions including braking and cruise control, a successful attack on the network would typically lead to the compromise of the entire system. Table 2.2 shows the differences between several common in-vehicle network protocols.

Protocol	Interface	Topology	Bandwidth	Transmission	Arbitration
CAN [250]	Multi-Master	Bus	1 Mbps	Asynchronous	Bit-wise arbitration where lowest message ID gets control of the bus.
LIN [228]	Master-Slave	Bus	20 Kbps	Synchronous	All messages are initiated by the master and one or zero slaves will respond to a given message.
MOST [102]	Timing Master-Slave	Daisy Chain or Star	150 Mbps	Synchronous	Access token is passed around the bus in a circle. A node can only transmit data if it has the token, ensuring fair access for all nodes.
FlexRay [182]	Multi-Master	Star or Bus	10 Mbps	Synchronous and Asynchronous	Static segment with fixed interval messages and dynamic segment with CAN-like arbitration.

Table 2.2: Comparison between several common in-vehicle network protocols.

Wolf [290] discusses many inherent vulnerabilities of in-vehicle networks. For example, LIN uses a master-slave architecture with all communication initiated by the master [228]. If the master is compromised then the entire sub-network of slave nodes can be disabled. Since LIN networks often control auxiliary features such as windows, lights, mirrors, fans,

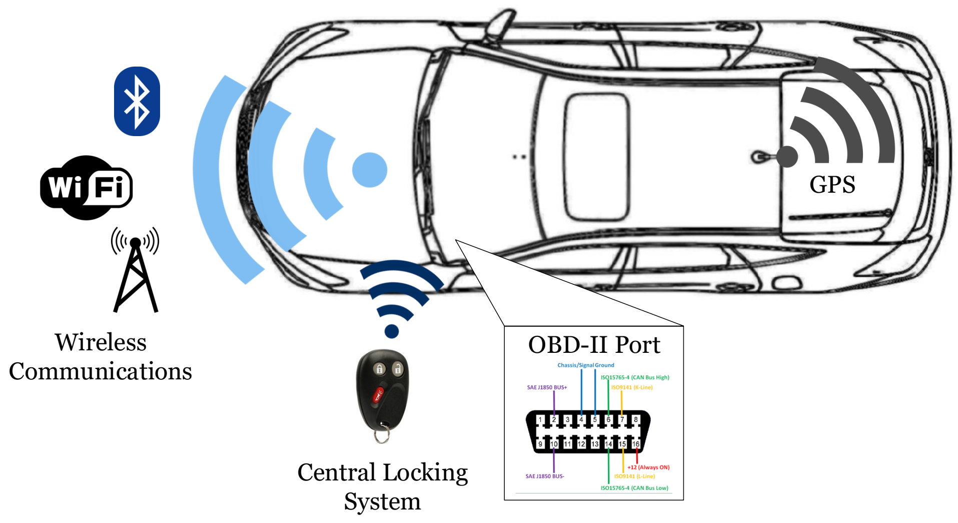


Figure 2.2: Common internal and remote attack vectors on automotive software systems are the diagnostic ports and telematics system.

etc., this can impact vehicle usability. Another example is the MOST protocol, which is primarily used for multimedia and infotainment. MOST networks use a ring or daisy-chain topology and have a single timing master node that continuously sends timing frames to synchronize slaves [102]. Since this is the only form of synchronization in the network, an attacker can send malicious timing frames to desynchronize nodes and disable the bus. This can render infotainment/telematics devices inoperable. An even greater vulnerability exists with CAN and FlexRay networks as they are designed for high speed, real-time systems and are often implemented in safety-critical applications. Both FlexRay and CAN are susceptible to exploits due to lack of authentication or encryption [182, 250]. This can allow attackers to cause malfunctions in safety-critical systems such as stability control, anti-lock braking, and engine management as well as in drive-by-wire systems such as electronic throttle and steering. This section primarily focuses on the security of CAN networks, since they have been ubiquitous and often form the primary communication bus in most vehicles. Many recent studies [51, 158, 85], have shown that it can be compromised by an attacker with relative ease, allowing them to enable or disable critical safety systems.

The two most common attacks for CAN-Bus exploitation are through (1) diagnostic ports and (2) telematics or infotainment systems. Figure 2.2 illustrates the different attack vectors. Diagnostic ports are a common entry point for an attacker due to the relative ease associated with launching an attack, assuming the attacker has physical access to the diagnostic port (i.e., access to the vehicle). Telematics and infotainment systems often use wireless protocols such as Bluetooth, cellular 2G/3G/4G, WiFi, and GPS, which enable attackers to remotely interface with these systems and launch attacks.

2.5.1 Attacks through Physical Access

In the United States, all vehicles sold since 1996 are required to use an OBD-II diagnostic port (specified in SAE J1962) to transmit emissions-related codes and data for vehicular emissions testing. Additionally, US legislation requires all vehicles sold since 2008 to support the ISO 15765-defined CAN standard through this OBD-II interface. Although the requirement is only for emissions-related information, most manufacturers use it as a primary diagnostic and reprogramming port as well. Since the port directly connects to several onboard computers via CAN, an attacker with physical access to the vehicle can easily launch attacks and compromise critical vehicle systems. The attacker could be an individual with legitimate access, (e.g., a valet driver or mechanic), or someone who gains illegitimate access e.g., through burglary. Once the attacker gains physical access, there is a wide array of OBD-II adapters available online to allow them to transmit and receive CAN messages.

Attacks through physical access, while easy to administer, have not been perceived as a "real" threat to automotive security. A standard response to such an attack has been that, if the attacker did have physical access, they could simply cut the brake wire or perform other similar damage, rather than hacking the vehicle through a CAN network. Nevertheless, as attention to automotive security has intensified in recent years, there have been efforts to mitigate such attacks. To combat exploits that utilize the physical OBD-II port, Markham *et al.* [183] proposed a role-based access control policy: each commercial OBD-II device would be certified by manufacturers and would send a public key and X.509 certificate to the vehicle to prove its identity. Once a device is verified by the vehicle, it is given access to certain systems based on its privilege; non-certified devices would only have permission to read the bus, while a certified mechanic's scan tool would have both read and write permissions. Nevertheless, note that attacks similar to the PassThru exploit [51] would circumvent this form of authentication.

2.5.2 Remote Attacks through Infotainment/Telematics

The Defense Advanced Research Projects Agency (DARPA) has demonstrated exploitable hacks in vehicular infotainment applications such as the UConnect® system in Chrysler, Jeep, FIAT, etc. [188]. They demonstrated that they could remotely control a vehicle via CAN bus commands. Their hacking demonstrations resulted in several recalls, including 1.4 million Chrysler automobiles [100].

Since 2004, the Environmental Protection Agency requires all vehicles manufactured in the US to support SAE J2534 PassThru devices, allowing Windows computers to communicate with a vehicle's internal bus networks. Consequently, many mechanics and technicians use J2534 PassThru devices for diagnostics and emissions testing. PassThru devices connect to the OBD-II port in vehicles and communicate with the Windows machine via a wired or wireless network. Checkoway et al. [51] showed how it is possible to hack these devices remotely through a local WiFi network. Since the PassThru device used by Checkoway depended on external network security, its communication over the local wireless network was not secured. This allowed them to perform a shell injection and install malicious binaries on the PassThru device and use the PassThru to install malicious code on a connected vehicle as well. Checkoway also demonstrated that a worm could be implemented to copy malicious code between multiple PassThru devices on the same network, increasing the impact of this attack. Other remote attacks include the exploitation of vulnerabilities in infotainment/telematics systems. These systems often include interfaces for Bluetooth, cellular, GPS, and other wireless protocols, as well as a communication channel to the internal CAN network; this makes them particularly attractive targets for remote attacks. Exploits to these systems often involve traditional hardware and software security exploits. For example, Checkoway *et al.* [51] showed a buffer overflow attack by installing a simple Trojan application on a connected Android phone; the application listened to Bluetooth traffic to determine if a certain model of telematics unit was connected and, if so, delivered the attack payload. Furthermore, using the bridging capability of the infotainment system, they could send arbitrary CAN messages to the internal CAN network.

2.5.3 Integrity and Availability Attacks

CAN was designed for real-time systems and prioritizes speed and reliability of delivery. CAN messages are broadcast to every node in the network, permitting anyone with bus access to perform packet sniffing. Additionally, CAN messages do not contain any authentication information to verify senders; the message ID is the only identifier used by a node to determine if it should process a message. This enables attackers to easily perform replay attacks by sending packets with message IDs that match the IDs of legitimate messages they want to spoof. Since CAN messages control various driving functions, attacks on integrity and availability can be mounted through appropriate CAN messages. For instance, Koscher *et al.* [158] showed how to send specific CAN messages in consumer vehicles that utilize electronic stability and brake control (e.g., ABS braking) to enable and disable brakes at speed.

It is difficult to prevent replay and availability attacks on CAN networks without significant protocol changes. However, there has been interest in detecting attacks non-intrusively by checking for anomalous bus traffic. Several intrusion detection strategies have been developed to defend against attacks on in-vehicle systems. Taylor *et al.* [259] demonstrated a non-intrusive anomaly detector for identifying replay attacks on CAN networks. The algorithm measures inter-packet timing over a sliding window and compares average times to historical averages to create an anomaly signal, and targets both replay and availability attacks. Note, however, that while such a solution is effective at detecting availability and replay attacks for periodic messages, they are ineffective at detecting attacks involving non-periodic messages due to their reliance on historic timing averages. Additionally, since these methods do not check message data for anomalies (only message timing), an attacker who can modify data within periodic messages without affecting timing intervals would be able to subvert these methods. Cho *et al.* [62] proposed an anomaly-based intrusion detection system (IDS) that utilizes the intervals and clock skews of periodic in-vehicle messages to create unique fingerprints for each ECU. Deviations from this signature indicate an intrusion into the network by a compromised ECU or other device. The proposed IDS was able to detect these intrusions with a false positive rate of 0.055%.

2.5.4 Authentication and Non-Repudiation Attacks

CAN networks have many restrictions that make it difficult to implement many known authentication protocols. Herrewege [272] discusses many of these restrictions. First, since CAN networks often have hard real-time constraints, one cannot introduce an authentication protocol that significantly impacts message timings. Second, each CAN message can only contain a maximum of 8 bytes, meaning that extra authentication data cannot simply be appended to existing messages. Third, since CAN message IDs correspond to specific functions, it is not feasible to add extra IDs for authentication data. Finally, the unidirectional message-passing methodology used by CAN makes it difficult to directly address specific nodes without using a rudimentary method such as flags. The combination of these factors makes it difficult for vehicle manufacturers to implement secure access control policies without significant time or capital investment. Car manufacturers typically prevent unauthorized re-flashing of the software on ECUs through CAN; however, the restrictions discussed above imply that only light-weight authentications can be implemented, which can be easily bypassed resulting in integrity and non-repudiation attacks through unauthorized, over-the-air update. Koscher *et al.* [158] showed that in a mid-range consumer vehicle, the authentication scheme used to control write access is a simple challenge-response pair: the car will ask for a 16-bit key which, if provided, unlocks the ECU software. They demonstrated that this form of key can be cracked with brute force in a matter of days.

Some manufacturers choose to keep critical systems on a separate, high-speed bus instead of on the primary bus so that critical systems are not affected if the primary bus is disabled. This can help prevent attacks on critical systems; however, since it is relatively easy for an attacker to re-flash ECUs, the attacker could compromise any ECU which communicates on both networks. For example, Koscher *et al.* [158] showed that the telematics unit (which communicates with both networks) was able to be reprogrammed from the low-speed bus to send custom messages on the high-speed bus.

Furthermore, Koscher *et al.* showed how to apply software reprogramming to launch non-repudiation attacks as follows. One can introduce a Trojan software by re-flashing the ECU such that the existing functionality would not be affected, allowing the original software and the malware to coexist. After the malware executes an attack (e.g., disabling the engine or locking the brakes), it would delete itself and relevant log data from the ECU to prevent detection during a forensic investigation.

Addressing the above attacks requires the development of authentication protocols that can meet CAN's real-time requirements. Herrewege *et al.* [272] presented a message authentication protocol named CANAuth, which inserts a hashed message authentication code (HMAC) between sampling points of a CAN bus interface. This is done using the CAN+ protocol proposed earlier by Ziermann [305] to encode data at a higher frequency within a single CAN bit without interfering with the underlying CAN bus protocol.

2.5.5 Ransomware and Thefts through CAN

The idea of a ransomware attack is to make a system functionality inaccessible to the user and demand ransom in exchange for returning access. In cybersecurity, this attack typically takes the form of encrypting important system files or locking functionality. In automotive systems, however, an attacker with the ability to send CAN messages can easily mount ransomware attacks by gaining control over a variety of in-vehicle functionalities. For instance, most vehicles today employ a central locking system and use CAN to control in-vehicle displays and user interfaces. Koscher *et al.* [158] showed that it is easy to control the locking of vehicle doors; turning on the horn; activating and deactivating the Heating, Ventilation, and Cooling (HVAC) system; or displaying arbitrary messages through the panel cluster display. In addition to ransomware, it is easy to use CAN to enable theft of the vehicle silently without activating the alarm, e.g., by sending messages to the telematics unit to successively (1) unlock the doors, (2) disable the immobilizer, and (3) start the engine.

2.6 Security of Vehicular Communications

A key feature of emergent autonomous vehicles is the ability to communicate with other vehicles, with the infrastructure, and with other devices connected to the Internet. Many features of autonomous transportation depend on such communications, including platooning, cooperative route management, etc. Consequently, there has been significant interest in designing effective vehicular communications. Thus, potential threats on such communications and the proposed defenses are observed in this section.

Vehicular communication or Vehicular-to-Everything (V2X) has been introduced as an amendment to the IEEE 802.11p standard. The standard was originally intended for Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) communications [4]. However, Vehicle-to-IoT (V2IoT) communications are anticipated to be implemented and standardized in the near future [16]. 5G-based Cellular-V2X is also being introduced by companies such as Qualcomm to compete against 802.11p as the leading V2X standard. 5G promises to be revolutionary for V2X due to its higher bandwidth capacity, smaller cell sizes, and new beam-forming capabilities relative to 4G Long-Term Evolution (LTE) [35, 180]. Figure 2.3 provides a visualization of the connected environment induced by V2X. A major part of security challenges in V2X is inherited from, and similar to, those in non-mobile ad hoc wireless networks. However, a compromise in V2X is much more serious since automobiles are safety-critical, electro-mechanical systems that influence major factors of peoples' lives [83]. For this reason, there have been efforts from standards bodies including IEEE and the European Telecommunications Standards Institute (ETSI) to develop standards and guidelines for V2X communication and Intelligent Transportation Systems (ITS) to meet the security objectives [4, 78, 289].

Unfortunately, due to the complexity of these systems and their subsystems, it is challenging to guarantee or even satisfy a majority of these security objectives [232]. Attack vectors targeting V2X are large and diverse as the methods used to compromise a vehicle's security depend on the kinds of entry points accessible to the attacker. V2X attacks are generally categorized by sophistication levels based on the distance between the attacker and the target vehicle. A direct/physical attack can be mounted by an attacker who is able to obtain physical access to the vehicle or hardware (e.g., OBU, CAN bus, transceivers), either as the owner of the vehicle or via successful attacks/exploits of the CAN network as discussed in Section 2.5. A remote attack is mounted by an attacker that does not have direct access to the vehicle.

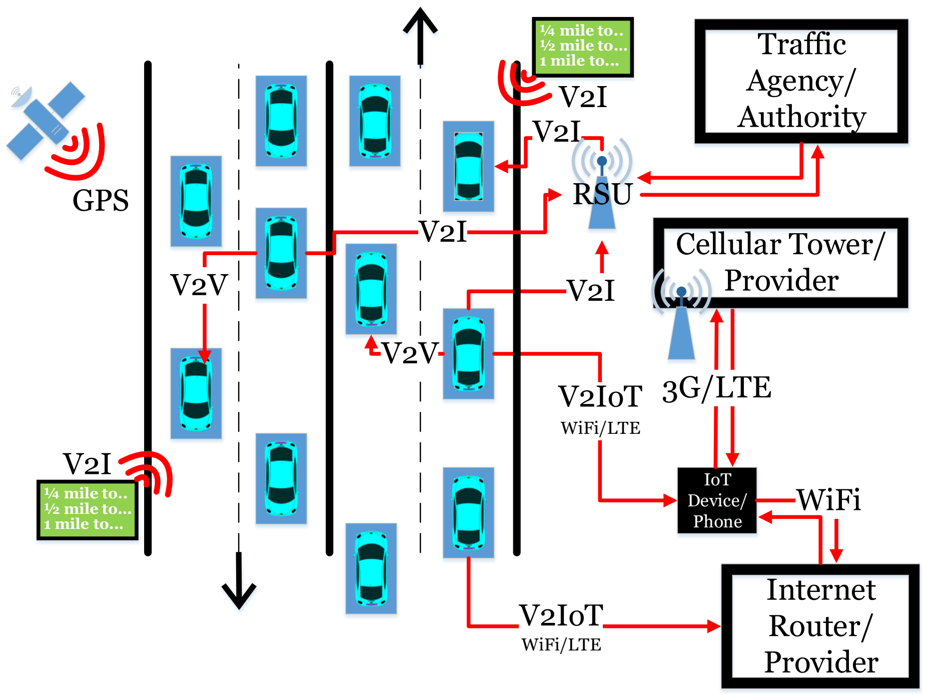


Figure 2.3: Vehicular communication aka Vehicle-to-Everything (V2X) environment with traditional, connected, and autonomous vehicles. Each line corresponds to a type of one or two-way communication channel for a specific application (V2V, V2I, V2IoT). Each connectivity line may also represent a potential attack vector for an exploitation.

2.6.1 Confidentiality Attacks

Confidentiality may be breached if an attacker directly accesses their OBU (as shown in the previous section) or purchases and implements 802.11p on a Software-Defined Radio (SDR) to sniff packets containing private or critical information [40]. For example, they could track a nearby vehicle via the position, speed, and action identities (unique to the event and contains information about originator) in V2V Decentralized Environmental Notification Messages (DENM) [232] or steal someone's credit card information transmitted over the air for Electronic Toll Collection (ETC) [166]. Tracking may lead to identifying a driver's behaviors, personal interests, home/work address, and/or their real identity [41, 126]. In the future, when peer-to-peer sharing is implemented, a key challenge will be to ensure the privacy of network traffic between vehicles, infrastructure, and IoT devices [169, 83].

To address the requirement for confidentiality, asymmetric key-based encryption methods have been proposed (e.g., Elliptical Curve Cryptography) by IEEE [4]. However, these methods are costly and challenging to implement in the ad hoc, heterogeneous environment of V2V communications [232]. Another challenge is latency, which must be kept to a minimum (less than or equal to 50ms for triggering events and resulting actions). Thus, ensuring that a cryptographic solution is both reliable and fast is a major challenge [232]. Furthermore, encryption methods should also be able to adapt to the situation (emergency or entertainment) to reduce energy and timing costs and ensure both safety and security. There have been proposed solutions [284, 283, 176] which attempt to use the benefits of the dynamically-changing physical environment to quickly generate highly random, symmetric cryptographic keys by adapting to the energy and timing constraints of V2X scenarios and the reciprocal fading properties of the wireless channel. Other solutions [76, 257, 292] attempt to algorithmically reduce the overhead of existing cryptographic solutions.

To prevent attacks on privacy, researchers have recommended using pseudonyms, sending data during only a part of the taken route (rather than all), ensuring k-anonymity, and consistently updating unique identifiers such as the MAC address, public key certificates, and probe message IDs [41, 173]. Some solutions provide domain-specific mitigation and prevention approaches for specific applications such as Electronic Toll Booth Collection [24, 185].

2.6.2 Integrity, Authentication and Non-Repudiation

Attacks on integrity and authentication typically involve tampering, fuzzing, and spoofing in some fashion. Tampering or fuzzing attacks involve the modification or injection of noise into packets sent over the air to confuse the involved parties, but they do not require attackers to masquerade as others. Besides remote V2X attacks, attackers may maliciously alter the code of in-vehicle CPUs, e.g., using malware or reflashing (via physical access to OBU or remote attacks to the telematics/infotainment from V2IoT [203]), or modify the original data before it is transmitted. Spoofing attacks, such as the Sybil attack, are detrimental to network productivity and breach both the integrity and authentication security objectives.

Douceur [75] proposed the Sybil attack. It involves a malicious node that adopts multiple addresses of legitimate (Sybil) nodes. This means that all messages will be rerouted to this malicious node instead of the legitimate nodes. Having these messages, the attacker can tamper with them and resend them to the legitimate nodes, or deceive nearby vehicles into believing that they are surrounded by traffic to get an empty route for itself once others choose alternate routes [75, 160]. Another attack method: tunneling, involves imitating a short wireless channel between two legitimate nodes from both ends of a network [214]. It causes the two legitimate nodes to select the malicious node in their routing algorithms. In turn, this allows the malicious node to infer information about the nodes, modify packets, and delay their communication attack availability. Timing-faking attacks were also shown to be effective against V2X systems. By delaying the timing of packets, RSUs will end up making incorrect decisions and force vehicles to enter sub-optimal routes with traffic, accidents, or other unforeseen circumstances [253].

Attacks on integrity and/or authentication could lead to a variety of impacts, including but not limited to traffic congestion and extra fuel costs, lower travel time for an attacker, ransomware, injury, and even murder. Garip *et al.* [91] showed how to simulate V2V attacks on connected autonomous vehicles using botnets (many bot vehicles in a targeted area). In their Manhattan grid experiments, they discovered that such attacks could overcome correlation-based defenses and cause traffic congestion (increase in average trip time by 50%) when only 1% of traffic is in the botnet area. Various recent researches [97, 220, 221, 56] discovered that traffic controllers were highly vulnerable to spoofing attacks. These attacks will lead to sub-optimal signal timing plans at intersections or freeway ramps to cause more traffic congestion, reduced travel times for attackers, or even accidents due to spoofed or tampered traffic signal information or unexpected timing changes and distracted/unaware drivers. Wireless authentication is also being implemented with Electric Vehicles (EVs) and the Smart Grid. In particular, there is a standardization where EVs would be able to use keys and certificates to wirelessly authenticate with a charging station and recharge the vehicle [49, 50]. However, an attacker (car thief) nearby may perform a substitution attack and use the victim's credentials instead of their own (which are invalid) to charge their vehicle.

Attacks that violate non-repudiation typically either directly target related security requirements (i.e., integrity, authentication, and availability) or directly target weak points in the non-repudiation schemes. For example, an attack may involve deciphering a weak cryptographic key used in a non-repudiation scheme (e.g., digital signature) or delaying mechanisms that verify the action history of a vehicle or node (e.g., voting, blockchain) [277, 71, 18].

Defending against attacks on vehicular communications is of crucial importance to the proliferation of connected vehicles. In the 802.11p/WAVE standard, the necessary protection mechanisms provided for integrity include using a Message Authentication Code (MAC) (if using symmetric keys) over the data, and a digital signature (if using asymmetric keys and identities) via RSUs and/or authorities like the Department of Motor Vehicles [110, 254, 173, 212, 63, 213, 232, 17]. These solutions may ensure authentication and non-repudiation as well. Combining these with a tamper-resistant cryptographic unit (such as the Trusted Platform Module in [104]) can provide significant protection against the attacks discussed above. Maintaining timeliness and freshness additionally requires time-variant parameters [232, 17]. However, note that these defenses may push development costs up and suffer from deterministic seeds, mismanagement of secret keys, and occasionally the tight resource constraints of embedded devices.

There has also been significant work on detection methods for integrity violations in vehicular communications. Relevant approaches include correlating messages from neighbors or using a reputation-based mechanism (via RSUs or authorities) to either infer the trustworthiness of messages or immediately detect tampered messages [38, 108, 91, 230, 45, 299]. Plausibility checks on the received data (time and location) have been proposed to prevent usage of spoofed data or even detect Sybil attacks (based on GPS data [107]). In [110], the authors propose a low-cost, position-based routing protocol using digital signatures/certificates, plausibility checks, and rate limitations to limit attacker capabilities. Another approach is to provide watchdog vehicles to monitor network traffic and identify potential attackers [116].

2.6.3 Availability

Since availability is intertwined with integrity and authentication, many of the attacks on integrity and non-repudiation discussed in Section 2.6.2 also impact availability. Furthermore, there are many networking attacks unique to availability, e.g., flooding/spamming, blackholes/greyholes/wormholes, physical layer jamming, and malware. Impacts of blackholes/greyholes/wormholes are studied in many recent chapters [36, 268, 282], which demonstrate attacks resulting in the network dropping packets in flight. Flooding and spamming attacks include message-based Denial of Service [39, 114]. Basciftci *et al.* [30] demonstrated a physical layer jamming attack with SDRs from National Instruments and simulated the same jamming attack in an LTE network simulation platform to cause a performance drop of over 40% for more than 50% of users. Finally, after malware injection into a vehicle, infrastructure, or IoT device, an attacker may purposely interfere with the receptions and processing of data to reduce the operational effectiveness of a vehicle and its peers.

Given the close correspondence between availability attacks and integration/repudiation attacks, defenses against the latter also serve as defenses against the former. However, there are also specific detection and prevention methods against availability attacks. Kaur *et al.* [147] present a detection and prevention technique for wormhole attacks by forcing authenticated nodes to hash their routing-based messages and also increment the number of hops appropriately for a unique decision message. If the hops were modified by the attacker, then the hash will be different than the hashed version of the legitimate message and the malicious message will be discarded. Khatoun *et al.* [152] provide a solution to identify malicious nodes performing Black Hole attacks by aggregating and analyzing information from RSUs and vehicles to measure the reliability and reputation of nodes. Jamming attacks may be mitigated or prevented via network coding techniques [94].

2.6.4 V2X and 5G

The upcoming complex 5G ecosystem is envisioned to include autonomous and connected vehicles, drones, air traffic control, transportation systems, health, smart factories, smart homes, smart cities, cloud-driven systems (robots and virtual reality), industrial processes and much more [175, 14, 204, 48, 179]. By 2020, it is expected that over 25 billion IoT units will be connected via various wireless and wired networking protocols of all types (automotive systems alone are expected to utilize 3G, 4G LTE, IEEE 802.11p, intra-networking, Bluetooth, and ZigBee among others) [234].

The 5G ecosystem promises exciting business opportunities, but its extreme level of interconnectivity is also a double-edged sword and comes with risks.

Due to the inter-connectivity of various devices under various protocols, the attack surface will be ever-growing and attractive for malicious entities and also terribly difficult for businesses to manage. Attacks may start from one endpoint to another endpoint in a completely different subsystem (e.g., smart home to connected autonomous vehicle). Devices and protocols that were once considered too complex for attackers to target or bother with are now more commonplace and well-understood by hackers.

In 2016, there was a leap in malware attacks [288, 115] and in 2017 alone, there was a 250% increase in mobile ransomware attacks due to the rapid adoption of LTE and IoT [234]. It is clear that when devices with legacy security solutions will become connected, unless properly secured, 5G devices will become attractive targets for larger-scale attacks such as the Mirai DDoS in 2016 [260, 21, 155]. The Mirai malware enabled attackers to seek out vulnerable devices via Telnet (incidentally Telnet was found to be potentially exploitable in several of the automotive aforementioned research works [158, 51, 190]) to take control over them, to prevent users from regaining control, and to utilize them to perform large-scale DDoS attacks on Internet service provider devices at the lower-layer Internet protocols [234, 260, 32].

It would not be surprising if many ECUs in vehicular networks were exploited to become a part of large scale botnet attacks (potentially up to the Tbps traffic volume scale [260]) or even the target of DoS attacks. Further, new types of networking protocols and applications resulting from 5G (e.g., Software-Defined Networks (SDN), Virtual Mobile Network Operators (VMNO), Mobile Edge Computing) reduce the gap and create softer boundaries between devices. Thus, they require new security designs and solutions to prevent access-related exploits.

In particular, infotainment systems of connected and autonomous vehicles will be connected to all sorts of devices for many services (e.g., entertainment [193], performance like battery management [179], and traffic control [83]). They will become attractive targets for threats such as ransomware or direct vehicle control. Finally, privacy concerns on identity tracking, behavior inferences, subscribed services, locations, and mobility patterns will rise because of the need for and capability to process massive data traffic flows through 5G [175, 32]. Such vital user information may be exploited in unethical ways or may be used in spoofing attacks. Battery Management System (BMS) security is of notable importance since BMSs and batteries are present in many types of devices and systems in the IoT and ITS ecosystems. Chapter 5 will go into further detail regarding BMS security.

Despite these potential security risks, the large scale, virtualization, and inherent distributive properties of future 5G networks may be also useful to eliminate threats such as DDoS attacks [12, 32]. Improvements in security techniques like firewalls, server load balancing, and FPGA-based Flexible Traffic Acceleration [234] and employment of physical layer security [283, 284, 295] will definitely help as well. In short, to address the risks of legacy software and hardware connecting with other devices through 5G, both businesses and service providers alike must strive to design their products with security in mind from the ground up.

2.7 Security of Vehicular Components

In addition to the communication mechanisms (whether in-vehicle or V2X), electronics components in the vehicle are also obviously subject to attacks.

2.7.1 Privacy Attacks on Infotainment Components

Most modern consumer vehicles have voice control or hands-free calling to allow users to keep their eyes on the road while using infotainment systems or making phone calls. Note that the microphone remains active for the entire duration of phone calls. These technologies can be exploited by an attacker to covertly record audio inside the vehicle. Checkoway *et al.* [51] demonstrated how to use a compromised telematics unit to record audio from an in-cabin microphone and stream it through a cellular network. Furthermore, vehicle location data can be extracted from the telematics unit as well, enabling attackers to monitor a user's location at all times. This could be used to identify high-value targets, such as owners of expensive vehicles who park at large corporations, to potentially find their home address for further surveillance or theft. The attacks on aftermarket TCUs demonstrated by Foster also facilitate these forms of data extraction [85].

2.7.2 Attacks on Wireless Key Entry and Ignition

Since the mid-1990s, Radio Frequency Identification (RFID), Remote Keyless Entry (RKE) and/or Remote Keyless Ignition (RKI) have commonly been implemented for consumer comfort and vehicle security against thieves. Ironically, these wireless communication-based solutions are also insecure. There have been several works [42, 247, 87] that found vulnerabilities in all three applications primarily because of design errors. Furthermore, stringent cost requirements make the implementation of advanced and sophisticated protection in these areas challenging.

In particular, signals to open or lock a car or start the engine could be stored, blocked, or relayed either wirelessly or over a cable. Such attacks could allow thieves to unlock and/or start a vehicle despite its owner being physically away. In 2005, a Texas Instrument RFID transponder used as an ignition key in millions of vehicles was also found to be hackable due to weak cryptographic keys [42]. In 2015, Kamkar developed and presented the RollJam attack on RKE [143]. The RollJam exploit simultaneously stores and jams a signal sent to unlock the door; then, when the driver sends an unlock signal to the door again, it is again blocked but the first stored signal is sent instead to the vehicle's receiver. This allows the attacker to use the second stored signal to unlock the car at will. Such an attack would only cost approximately $32 and it was successfully tested on Nissan, Cadillac, Ford, Toyota, Lotus, Volkswagen, and Chrysler vehicles.

Ibrahim *et al.* [127] demonstrated a three-step attack involving (1) set-up, (2) jamming and recording, and (3) hijacking. Their attacks were more flexible than the RollJam attack (remotely controller jammer, no need for precisely tuned jammer) and easier (constant jamming forces user to eventually use mechanical key and not reset the RKE code). They tested their attack with various distance parameters (distance from user to vehicle and distance from user to attacker's logger device) on six vehicles Škoda Yeti (2016), Škoda Octavia (2009), Mazda 6 (2009), Toyota Rav4 (2014), Mitsubishi Pajero (2015) and Nissan Sunny (2014).

Note that the challenges to securing wireless key entry and ignition systems include resource limitations of hardware and the high costs of cryptographic solutions. Most researcher recommendations include using RF signal properties to verify if a user is truly nearby or not [153, 202, 233]. Yang *et al.* [296] propose a low cost (memory and complexity) solution that involves a challenge-response protocol based on distance bounding, where the verifier measures an upper-bound of the actual distance to the prover so that the attacker cannot convince them that they are closer than they really are. Furthermore, a possible solution to wireless attacks on wireless authentication between EVs and the smart grid has been recommended through a "cyber-physical authentication protocol" which requires physical access of the charging cable to verify the identity and legitimacy of a vehicle [50].

2.7.3 Sensor Attacks

Integrated and embedded sensors in automotive systems are crucial for the operation of connected and autonomous vehicles. With wireless communication, connected vehicles can exchange sensor data with each other for smarter applications and better control. Vehicles with autonomous capability need more informative and accurate sensors (e.g., LIDAR and camera), and more efficient and reliable algorithms for control (e.g., machine learning models) [171, 126]. Consequently, the security of sensors is critical to prevent severe functional and safety-critical impacts from exploits. Unfortunately, these sensors and sensor data-based algorithms are heavily vulnerable to malicious environmental and wireless communication modifications.

Rouf *et al.* developed an attack with a low-cost software-defined radio (SDR) that captured and read Tire Pressure Monitoring System (TPMS) communication packets from a vehicle up to 40 meters away [129]. TPMS messages also include identifiers of tire sensors that are sufficiently unique for attackers to use to track the vehicle. Furthermore, they demonstrated the possibility of injecting packets into the TPMS network to trigger a fake warning signal [129]. Several of the hacking works mentioned in Table 2.1 have experimented and demonstrated TPMS remote attacks on their testbeds.

There have also been studies of attacks on navigation systems [124, 123, 151]. These studies found that the GPS receiver was vulnerable to spoofing. Spoofing attacks would provide false location information and may lead to longer trip times or, worse, accidents. Correspondingly, radar, another sensory component used to measure distances, is also susceptible to jamming and spoofing [294]. Furthermore, recent research [209, 109] discovered that lasers or similar technology could spoof the existence of vehicles to LIDAR. Image-based machine learning algorithms and models can also be fooled to make incorrect and life-endangering decisions if small modifications were made to road signs or lines (e.g., stickers, markings, delineation) [294, 209, 70, 13, 64, 211, 245, 8, 122].

In addition to the above, given the complexity of autonomous and connected vehicles, there is a strong necessity to have miscellaneous sensors everywhere to ensure safety and performance. Examples include gyroscopes and anti-lock braking system (ABS) sensors as well as visible light, infrared, thermal infrared, odometric (accelerometers, gyroscopes, etc.), and acoustic sensors. Attacks on these types of sensors vary in difficulty because of their varying accessibility levels [208, 126]. ABS sensors could be spoofed or jammed via an electromagnetic actuator that can be as far as three meters away from the wheel speed sensors [238]. Visible light, infrared, and thermal infrared sensors can all be deceived or jammed with environment-based injections of the same medium (but attacks may be difficult due to limited ranges) [208, 226]. Magnetic or thermal attacks may potentially affect odometric sensors to affect vehicle navigation but the success probability is low due to the hardware costs, range, and timing of such attacks.

For attacks that attempt to deceive the sensor-based algorithms with changes in the environment, Yan *et al.* [294] applied redundancy, logic checking, confidence priority, and attack detection along with sensor fusion. Petit *et al.* [209] applied low-cost software solutions such as random sampling, multiple sampling (for LIDAR), and a shortened pulse period. For eavesdropping and data spoofing attacks on network-based sensors such as TPMS, typical mitigation approaches include better logic consistency checks and low-cost cryptographic solutions with freshness and a strong source of randomness to prevent unauthorized access or usage of fake data [232, 129].

GPS and GNSS spoofing may be prevented with Navigation Message Authentication (NMA) and replay/spoofing detection methods [123]. Assuming a multi-antenna array is being used, Danesnmand *et al.* proposed a low-cost method that first detects spoofing signals, maximizes authentic signals and then attenuates the spoofing signals [69]. Another approach to detect and defend against deception-based attacks on sensors and their algorithms is to perform a design-time and run-time secure state estimation and identify which sensors are trustworthy through satisfiability solving [242].

2.7.4 Attacks on Map-Based Navigation

Map-based navigation is an application that both connected and autonomous vehicles may utilize to reduce trip time and improve passenger comfort. Map data may be stored already in the vehicle, received from RSUs, or received from cloud-based and mobile-based applications such as Waze and Google Maps [132]. Attacks that poison these maps in storage or deceive navigation applications with ghost cars may lead to less effective navigation by vehicles and eventually, traffic congestion [298, 240].

A summary of the notable works mentioned in Sections 2.5-2.7 is summarized in Table 2.3 for works on Attack Methods and in Table 2.4 for works on Defense Methods.

2.8 Digging Deeper: A Car Hacking Case Study

The preceding sections attempted to provide a structure and taxonomy to the diversity of attack surfaces on current and emergent automotive systems, and the corresponding defenses. Nevertheless, successful attacks demonstrated on automotive systems actually cross-cut many of these structures. This section dives deeper into a specific, demonstrated attack on a modern automobile, , Miller and Valasek's 2015 exploitation of a Jeep Grand Cherokee. The authors have described this attack in detail in a white chapter [190], enabling researchers to identify the various vulnerabilities exploited to successfully compromise a deployed vehicle and get control over its functionality. These details make this work a good target for a pedagogical case study. In addition, key elements of this attack and some of the high-level insights are discussed. Readers interested in further understanding are referred to their white chapter.

The Miller-Valasek work was done in the backdrop of two previous works, by Koscher *et al.* [158] and Checkoway *et al.* [51]. Koscher *et al.*'s work showed that once an attacker can send CAN messages, they can easily control driving functionality; however, no means were provided for getting access to CAN messages remotely. Of course, an attacker with physical access to a victim's car can cause physical damage in other forms (eg, by cutting the brake

Year	Application	Attack Method	Attack Impact	Complexity	Citations
2010	In-Vehicle	Physical Access and Code Exploits	Vehicle Control	Low-High	[158]
2011	In-Vehicle	Remote Access and Code Exploits	Vehicle Control	Low-High	[42, 143, 127, 51]
2011	V2V / V2I	Timing Faking Attack	Suboptimal Routes/Traffic Congestion/Accidents	Low	[253]
2012	V2X	DDOS Blackhole Attack by Synchronization	Slightly Reduced Network Performance	Low-Medium	[39]
2015	V2V	Botnet-based Spoofing Attack	Suboptimal Routes/Traffic Congestion	Medium	[91]
2015	V2X	Greyhole Attack	Extremely Reduced Network Performance	Low	[282]
2015	V2X	Physical Layer Jamming	Slightly-Extremely Reduced Network Performance	Medium-High	[30]
2015	Autonomous Navigation	Camera and LIDAR Deception	Slightly-Extreme Reduced Network Performance	Low-High	[209, 211, 298]
2018	V2I and ITS	Data Spoofing to Traffic Controller or Traffic Sensors	Traffic Congestion	Medium	[56, 97, 220, 221, 298]
2018	Autonomous Navigation	Deception with Toxic Signs	Control and Performance Loss, Life-Endangering Situations	Medium	[291, 64, 245, 13, 211]
2008, 2015, 2018	Autonomous Navigation / Route Adaptation	GPS Spoofing	Control and Performance Loss, Life-Endangering Situations	Medium	[124, 120, 300]

Table 2.3: Notable works according to targeted applications, attack methods, descriptions, and complexity. Complexity is subjectively defined based on the requirements to launch the attack: deployment and resources (time, memory, space). There are three levels (low, medium, high), where one or more levels (ranges) are provided per attack category.

Application	Defense	Description	Complexity	Citations
V2X	Voting Architecture	Multi Agent-based voting	Medium-High	[71]
V2X	Blockchain Architecture	Verification of Authenticity, Integrity and Non-Repudiation	High	[45, 243, 293]
V2X	Certificate Management, Digital Signatures, and Message Authentication Codes	Verification of Authenticity and Integrity of Data	Medium-High	[254, 128, 173, 232, 17]
V2X	Low Cost Cryptographic Schemes	Physical Layer Key Generation and Exchange, ID-Based Cryptosystem	Low	[232, 284, 283, 63]
V2X	Watchdogs and Data-Based Malicious Behavior Detection	Spoofing/Sybil/Black-Hole/Gray-Hole Detection, Position-based Routing	Medium-High	[54, 207, 99, 108, 116, 304, 299, 230, 147, 152, 110]
V2X	Location Privacy Preservation and Pseudonyms	Pseudonyms based on ID and Context	Medium-High	[24, 37, 185, 196, 82]
V2X	Network Coding	Jamming Mitigation or Prevention	Medium-High	[94]
V2X	Vehicular Visible Light Communication, Infrared, Radar	Reduced range of connection	Medium-High	[271, 174, 244]
In-Vehicle and V2X	Malware Detection	Cloud-based On-Board Malware Defense Manager	High	[303]
In-Vehicle	Intrusion Detection	Timing-Based CAN-Bus Anomaly Detector	Low	[259]
In-Vehicle	Intrusion Detection	Clock-Based ECU Fingerprinting	Low	[62]
In-Vehicle	Role-Based Access Control	Authentication of Physical OBD-II Devices with X.509 Certificates	Medium-High	[183]
In-Vehicle	Authentication and Access Control	Encoding HMAC on top of Existing CAN-Bus Messages	Medium-High	[272]

Table 2.4: Notable works on defenses, their descriptions, and complexity. Complexity is subjectively defined based on the requirements to launch the defense: deployment and resources (time, memory, space). There are three levels (low, medium, high), where one or more levels (ranges) are provided per defense category.

wire). Consequently, while this demonstration was interesting, it was less than compelling. Checkoway *et al.* reported the ability to get remote access to CAN, but no details were provided. Miller and Valasek's chapter described the first compelling remote exploitation on a deployed vehicle with sufficient detail for the attack to be reproducible.

The hack proceeds in three key stages: (1) compromising the head unit, (2) identifying a pathway for access to CAN from the head unit, (3) message injection into CAN to compromise driving functionality. Note that each stage is non-trivial, eg CAN message injection requires significant analysis of CAN messages.

Compromising the Head Unit: The key idea behind this attack is to exploit a vulnerability in inter-process communication (IPC). IPC is a standard means for software processes to communicate with each other, either through standardized or proprietary protocols: the typical approach is for IPC services to be implemented through software *daemons* that use a variety of "sockets" to enable communication among processes. The IPC daemon in the Grand Cherokee was a standard daemon called D-Bus, which is a highly configurable IPC daemon used in a variety of embedded systems. Typically, communication through D-Bus requires authentication. The vulnerability exploited was an open, unmonitored port in D-Bus, that enabled anonymous access. Consequently, any entity or process that could connect to that port would be provided access to the D-bus services. In particular, if a hacker could get into the wireless network of a vehicle, then, given the knowledge of the port number for the specific open port, they could anonymously connect to D-Bus without requiring further authentication.

Getting Into the Vehicle Network: Obviously, the network of a car would not be open to public access, they are typically protected by a firewall. One option is for the hacker to physically hack into a connected electronic component inside the vehicle that is connected to the car's network; however, that would require physical access to the car. This problem was circumvented by exploiting another "feature" of the Jeep Grand Cherokee: the ability to connect from any device subscribed to the network of the car's wireless carrier. In particular, the network carrier for the cellular modem in the Jeep's head unit was Sprint, and this carrier provided a feature that enabled any Sprint device to communicate with any other Sprint device through the Sprint$^{\text{TM}}$ wireless network. Consequently, it was possible to get a Sprint burner phone, tether it to any computer, and thereby give that computer access to the Sprint network where the address and port of the victim D-Bus daemon was visible.

CAN Message Injection: Given the above steps, it became possible to remotely compromise the head unit. This enabled the attacker to have full control over the in-car infotainment including radio volume, temperature control, and the heads-up display among others. However, there was no direct connection between the head unit and the driving functionality of the car.[1] Achieving that would require the ability to inject arbitrary messages on the CAN-C bus that controlled the various ADAS components.

Obviously, the head unit components were not directly connected to this bus. However, they could not be physically isolated either, since many features in the car require communication between ADAS and infotainment, e.g., the ability to see the trajectory in the display while reversing, a feature available in most modern cars, would require communication of the angular momentum information from the wheels to the display component. To address this, the head unit includes two integrated circuits, an ARM$^{\text{TM}}$ and a V850$^{\text{TM}}$ with different components connected. The ARM component to which the radio was connected was not permitted to send CAN messages; the V850 could send CAN messages but was not directly connected to outside connections (and consequently, compromised head unit components). However, they were connected through an SPI link that enables communication between the two processors and, furthermore, the ARM processor could reprogram the V850 system through software. This enables the hacker to use a compromised ARM processor (through

[1]Through the control of the infotainment, they could show a speed on the display that was different from the actual speed of the car, but they could not actually affect the speed of the car.

exploitation of the head unit) to reprogram the V850 to accept any command provided through the SPI link. Consequently, any subsequent communication from ARM (e.g., CAN messages representing directives to brake, steering, accelerator, etc.) would be accepted by V850 and passed on through the CAN-C network to the appropriate component, completing the compromise.

It should be mentioned that this compromise is not possible on today's on-road vehicles (which have been patched by Jeep and Sprint), e.g., Sprint has blocked all the traffic to the exposed D-bus port thereby preventing the attacker from gaining access to the head unit of the vehicle over the internet. Nevertheless, it is worth noting the cost: Jeep had to recall over 1.4 million vehicles in response to the hack. From the perspective of this chapter, it is also important to realize that a practical hack of a car actually cuts across the developed taxonomy and typically involves multiple compromises. It is frightening that it is reasonably easy to perform such a hack on a deployed automotive system.

2.9 Automotive Security Validation in Practice

Given the plethora of attacks as discussed above, how does the automotive industry approach the security validation of current (and emergent) vehicles? Unfortunately, the state of the art today is primarily manual. In particular, much of the practice is based on *penetration testing* [219], i.e., performing security attacks on the car in a controlled environment. The Miller-Valasek hack discussed in Section 2.8 provides a blueprint of the approach that can be taken to approach this complex task. More generally, penetration testing for an automotive system typically involves three steps, e.g., finding an entry point, exploring and reverse-engineering various firmware code installed in the system, and finally identifying vulnerabilities in the firmware to gain control over the vehicle functionality. This section provides a brief insight into the process.

2.9.1 Finding Entry Points in a Vehicle

Physical access to a car can provide a multitude of entry points for a security hacker, e.g., through access to CAN via an OBD-II connector. However, as discussed in the preceding sections, it is possible to get remote access to the car. In fact, every external input to the car is typically explored as a potential entry point. In particular, most modern cars connect to the Internet through a device with cellular connectivity, such as a mobile phone. If this connection is through another device, e.g., via tethering with a mobile phone, that device becomes a point of vulnerability. If the car connects directly, e.g., through a cellular chip, it is more complex to find an entry point. One potential area of exploration include possible open ports that are sometimes accessible through the Internet.

Another area is the variety of security certificates, e.g., if the car connects through a secure socket layer (SSL), then that includes a variety of certificates, many of which include several configurations which can lead to the possibility of a misconfigured certificate. A third possibility is through a variety of remote commands. In particular, various vehicle functionalities can be accessed through mobile apps, e.g., remote engine start, remote door lock/unlock, remote climate control, etc. These commands typically use a middleware service e.g., an MQTT broker [3], which can provide an entry point for an attack as well.

The above techniques provide some obvious "low-hanging fruits" for penetration testing, and are often successful. However, the Internet connectivity of most vehicles is generally more secure. In particular, most electronic components of a car are typically protected by a firewall and not accessible externally through the Internet. Accessing such components requires first getting access to a computer within the vehicle's network. The Miller-Valasek work showed one way to do this, e.g., through the "feature" provided by the network provider Sprint that enabled any Sprint device to communicate with any other device within the Spring network, including devices that were within a vehicle. One avenue is to hack into one device of a car (possibly locally) and use the hacked device to enable access to other devices connected to the same network.

Such attacks can be thwarted by not permitting devices in one vehicle to access devices in other vehicles even within the same network. When this is implemented, various other techniques can be used, e.g., by implementing a cellular tower simulator/emulator, or applying fuzzing techniques [256] on the various external-facing software including the Bluetooth stack, USB stack, etc. In addition, many vehicles have a variety of exposed hardware interfaces including debug interfaces (e.g., JTAG), serial consoles, etc. Serial consoles are used during the development phase but sometimes inadvertently left open at deployment, sometimes including shells with root privilege. Finally, one can find entry points by observing or injecting CAN messages, e.g., sometimes it is possible to reprogram a CPU through CAN messages as demonstrated in Miller-Valasek.

2.9.2 Obtaining and Reverse-engineering Firmware

Simply finding an entry point is not sufficient to compromise a vehicle: one must also find ways to modify its functionality. This is typically performed through reverse-engineering and modifying the firmware. Sometimes the original firmware binary is directly available from the manufacturer's Web site or through an insider in the dealership. If not, the firmware can sometimes be lifted directly from the flash memory, or a root shell in the serial console. Once the firmware is obtained, various standard reverse-engineering tools can be applied to interpret the firmware, e.g., Binwalk [142], Ida Pro [1], etc. Note that this is not a trivial matter, e.g., firmware is not structured, and is often targeted for a variety of non-standard instruction set architectures (e.g., V850) with complex memory layout. Furthermore, they are large, e.g., of the order of Gigabytes in many cases. So it is not feasible to comprehend the entire functionality. However, it is often possible to short-change the process by identifying and interpreting specific functions or symbols. In particular, strings and symbols are sometimes left in the firmware which can provide convenient starting points for reverse-engineering.

2.9.3 Privilege Escalation

The final step in a successful automotive hack is the ability to run arbitrary code or send arbitrary messages. In traditional computing systems, most software processes do not have this privilege. However, in embedded devices, it is often common for all software processes to run with administrator privilege. Since much of automotive software has been derived from embedded systems both in design philosophy and in implementation, this feature is often available there as well. Even if all processes do not have administrator privilege, most boot-up processes do and are obvious targets for a hack. Other processes to target are IPC daemons, e.g., DBus, as demonstrated by Miller-Valasek. Finally, most automotive systems include processes for over-the-air firmware upgrades. These processes of necessity execute with administrator privilege and also have significant ability to control and modify the installed firmware; if such a process can be compromised, the hacker can exert significant control over the entire vehicular firmware.

2.9.4 Applicability of Formal Methods

Clearly, the above approaches are purely manual attacks, depending on deep human insight. It is certainly reasonable to ask why there are no systematic approaches to perform such hacks in today's practice. The answer to that question is complex, relating to unavailability and limited scalability of tools, and difficulty to integrate the tools into the complex security validation methodology. To illustrate this point, an example of *one* promising approach, e.g., formal methods, is taken. Formal methods entail the use of mathematical reasoning to identify errors or vulnerabilities in design. In principle, it is very attractive since unlike the approaches discussed above it can provide a mathematical guarantee on the robustness of a system component or find corner-case vulnerabilities that are difficult to exercise otherwise.

However, while there has been significant work on the use of formal methods for hardware and system security [219, 57], the application on automotive system-level security verification in practice is limited. There are several reasons for this. First, while there is some functional specification, a comprehensive specification of a vehicle functionality at the level of detail necessary for the applicability of formal methods is lacking. Second, even if available, such a specification would be extremely complex: even more pertinent, the implementation of an overall automotive system is extremely complicated with several cross-cutting modules related to hardware, software, and communication, together with both digital and analog/mixed-signal components.

Automated formal methods such as model checking, has been used in some targeted applications for verifying functional safety in individual automotive parts, e.g., individual SoC designs within an ECU [57], but it is difficult to scale such approaches beyond that level. Furthermore, a significant amount of collateral is missing at design time, e.g., fuse configurations necessary for ensuring life cycle isolation are not available during the RTL design of automotive hardware where formal methods could be applicable.

The above is not to discourage the extremely important role formal methods can play in security. In fact, a greater application would only be welcome. However, for any methodology, it is important to understand its shortcomings in order to find potential areas for improvement. Perhaps one way for formal methods to become applicable is through a better design management process, e.g., by ensuring all specification and other validation collateral are available at the time necessary, and models are available at different levels of abstraction (perhaps through automatic abstraction) to enable applicability of formal methods at the vehicle level.

2.10 Security of Automotive Supply Chain

The discussion in this chapter focused on automotive security at the vehicle level. Another component of automotive security entails vulnerabilities in the complex, rich supply chain involved in automotive electronics production. The supply chain of automotive electronics is complex, with several potential vulnerabilities. This section briefly gives a flavor of the challenges involved for the sake of completeness of the presentation. For further details, the reader is referred to a recent chapter [218] that exclusively addresses this subject.

Consider an electronic part developed for an automotive system. Typically, it would be developed by some electronic part vendor, and go through several tiers of part suppliers, eventually to an automotive manufacturer who would integrate the part into an automobile. Each player in this process can introduce several sensitive assets to the part. These include cryptographic keys, Digital Rights Management (DRM) keys (for infotainment parts), proprietary firmware, etc. [219]. Furthermore, each player in the supply chain can also include rogue or malicious agents, e.g., a rogue employee, a malicious CAD tool, or even an untrusted foundry. It is critical to ensure that the system is robust against attacks by such players.

Supply chain security has been an active topic of research in the hardware security community, with several excellent treatises [261, 33, 192, 34]. The security threats considered in the literature include Trojan insertions, IP piracy, cloning, counterfeit ICs, and overproduction, among others. All of these threats carry over to the automotive systems as well. However, automotive systems carry their own supply chain challenges. In particular, assets should be protected, not only after the system is in-field but also when it is with the subsequent players in the supply chain.

1. Assets introduced by the vendor should not be accessible to suppliers, automotive manufacturers, or end-users.

2. Assets introduced by a supplier or automotive manufacturer should not be accessible to any other party, including the original vendor.

3. All assets should be protected against side-channel attacks (e.g., voltage, temperature, or clock glitch attacks).

4. Customer and third-party software should be protected against unauthorized access.

To exacerbate the problem, note that a part may return to the original part vendor, e.g., after a field return. At this point, the part includes assets from all *subsequent* players in the supply chain which must be protected from the vendor. Additional sources of complexity arise from the fact that assets may be sprinkled across different parts, and cross-cut hardware, firmware, and software; furthermore, calls are not statically provisioned but may be created on the fly as the system executes. Finally, note that test and debug interfaces add to the vulnerabilities.

These interfaces provide the user with structured access to internal architectural and design features (e.g., scan chain, various design-for-debug features, etc.) for functional verification, manufacturing tests, and other related activities. Since these activities entail observability of internal states of the design (and consequently of assets stored), a key challenge is to ensure the testability of the part while preventing unauthorized access to assets. Finally, note that once a part moves from one player to another in the supply chain (referred to as a change in the "life cycle"), asset protection is adjusted through a configuration of fuses. On the other hand, fuse programming is performed through the use of the debug interface. This creates a circular dependency between the ability to program through this interface and the effect of the programming on the interface (e.g., a life cycle change may change the way the debug interface is accessed).

Addressing the problems above in today's practice is primarily manual, based on the expertise of experienced architects and designers. However, with the growing complexity of automotive electronics, this practice is getting increasingly difficult to implement, with numerous bugs and vulnerabilities found late in production or even after deployment. Currently, the industry is exploring trust provisioning schemes to address this problem at the architecture level: the idea in which assets are provisioned by various stake-holders through a specific, centralized trust model.

The trust model is typically defined by the supplier of the part who is also responsible for the architecture that enables various stake-holders to insert assets at different life cycle stages. The provisioning mechanism guarantees that (1) a service that does not need an asset does not get access to it, and (2) access and update to each asset satisfies the trust model. However, the approach is in infancy and its viability is not currently well understood. Furthermore, validation schemes have been explored to validate fuse configurations to ensure life cycle isolation, e.g., through formal methods.

2.11 Summary

This chapter has provided a summary and taxonomy of security challenges in current and emergent vehicles. Because the area is vast, the goal of this chapter is to provide a structure to the plethora of attacks that are known today and the defenses that have been considered for such attacks. This is a contribution to the ITS security community as a useful starting point for researchers and practitioners in the area.

Of course, the attack surface will increase significantly as we move towards self-driving, autonomous vehicles. Nevertheless, they are coming soon, e.g., major companies such as Waymo and Uber are moving towards launching close to 100,000 self-driving cars with Level 4 automation in various urban cities by 2021-22. The test vehicles are being deployed in different scenarios and have driven several millions of miles to cover the corner cases

[168]. They will include advanced sensors such as Lidar, radar, etc. and multiple cameras. The computational resources needed for a self-driving car are also significantly higher to accommodate the processing of large amounts of sensor data. This, in turn, necessitates modifications to the conventional firmware and software running on the underlying ECUs. Additionally, there will be a paradigm shift from cars being viewed only as production vehicles to widespread use as autonomous ride-sharing vehicles [161].

It is unknown how all these issues will truly affect the security of these vehicles. Furthermore, the increased connectivity of vehicles and infrastructure will broaden the potential attack surface significantly. The implementation of 5G and 802.11p connectivity enables unforeseen adversarial attacks not just on vehicles, but on infrastructure, traffic, and any Internet-connected systems. The security of V2X communication has not been studied in-depth, and regrettably, the industry currently does not consider security a priority for V2X development. Since V2X technology is in its infancy, it would be prudent for researchers to focus on methods to verify the security of vehicular communication systems before they become ubiquitous.

In spite of the above, it is not all “doom and gloom”. There are certain unique characteristic features of a ride-sharing autonomous vehicle that are inherently beneficial for security [44]. For instance, the communication modules are highly customized to be locked down as there is no requirement for providing different user interfaces through Bluetooth, Web browsers, etc., unlike the production vehicles. In particular, the head unit comprising of the telematics is no longer required in a ride-sharing vehicle, and the OBD-II port can also be locked in a non-standard location. Finally, there is a certain amount of obscurity in the architecture which makes it difficult for the user to gain significant access to the vehicle, extract the firmware, and test for exploits. Nevertheless, the security of these vehicles is a critical threat and it is crucial to comprehend, analyze, and mitigate security challenges in such vehicles.

Chapter 3

Security Analysis of Fixed Time Connected Traffic Control Systems

3.1 Introduction

3.1.1 Motivation

In recent years, it has been found that modern traffic control wireless networks are vulnerable to sensor tampering and cyber attacks due to a lack of attention in their security (no attack detection/prevention, weak or no encryption/authentication) during the manufacturing and installation processes [47, 96]. Besides experimental studies, there are real attacks such as the train control network hack by a Polish teen in 2008 causing fthe trams to derail [170] and the insider attacks on the traffic control network at a major Los Angeles intersection [286].

Although at first glance these security weaknesses are fixable and may appear to not represent all traffic control systems, it is generally agreed upon that some present and future vulnerabilities cannot be prevented because these systems require long life times, have complicated software-upgrading methods, and are becoming more interconnected with other devices and the Internet each day [164, 302]. In fact, in 2014, a whopping 15000+ existing wireless devices (sensors and controller) deployed across 45 U.S. states and 10 countries were found to be potentially vulnerable to exploits that could lead to remote modifications of traffic timing control. Despite such a serious finding, the sensor's vendor ignored the report at the time. Such a negative reaction clearly demonstrates the high chance of existence of exploitable vulnerabilities in similar and/or other devices in traffic control systems [46].

The objectives of attackers may vary, but in general they involve some form of congestion since they are prevented from forcing the system into an unsafe state to directly cause accidents (due to hardware fail-safes against remote modifications or easy detection of physical tampering). An attacker may be a person with a personal vendetta who wants to slow down the travel time of their enemy/enemies, or perhaps one or more members of a company/organization that desire to reduce traffic for their benefits and/or increase it against competitors. They may even be a recreational hacker interested in causing trouble for fun or a malicious one determined to cause havoc for major cities.

Existing works on ITS security include case studies on traffic control wireless networks [96, 47], connected and autonomous cars [79, 264, 55, 52, 121, 284], ride-sharing applications [298] and sensors in freeway traffic control systems [222, 223]. Works [96, 47] have demonstrated that networked traffic systems are vulnerable to a variety of cyber attacks. Other works [95, 164, 223, 55, 241] attempt to quantify cyber-physical impacts of traffic control cyber-attacks and also attempt to solve for optimal attacks. Shoukry et al. (2018) also tries to design a traffic network that is resilient to Sybil (fake car) attacks. These works demonstrate that attacks on traffic control systems should not be underestimated and that it is critical to research how to formalize these attacks and their impacts. Tools should be developed to effectively aid traffic designers and engineers in the secure design and maintenance of these systems.

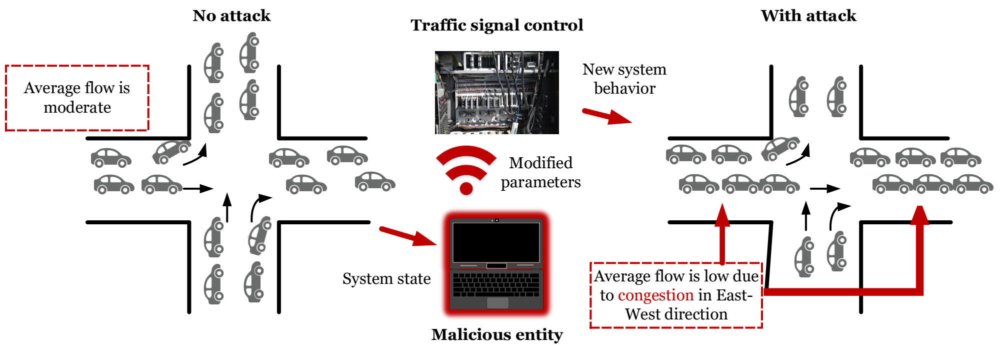


Figure 3.1: Motivational example of the potential effects (congestion in one or both directions) due to malicious modifications on traffic signal control settings of ITS.

3.1.2 Motivational Example

Figure 3.1 illustrates a motivational example where a fixed-timing traffic controller is hacked and the malicious entity modifies the signal control settings (this is explained more in Section 3.3). The network model is composed of two intersecting one-way roads, where cars on one road can only cross the intersection when it is their respective signal phase (two signal phases). The modifications include a reduction of green time for the first phase and an increase for the second phase. This attack causes congestion, a reduction in average speed and traffic flow (like that in the right side of Figure 3.1), and eventually gridlock, where the average traffic flow is equal to zero. If a traffic management agency detects the changes after several cycles and reverts the system settings to how they originally were, the long-term impact of the attack may still remain if the attacker carefully chose the modifications and attack timing.

3.1.3 Research Challenges and Contributions

To model similar attacks to the one in the motivational example, related works [55, 95, 164] have implemented existing traffic network and behavior models such as the Cell Transmission Model (CTM) [68] and the single queue model-based network model [274]. Still, most of these models contain limited analytical properties, miss certain behaviors, and are challenging to use for larger grid networks (e.g., Manhattan grid networks). To address these challenges, the system model proposed in [89] is used. This system model includes the Link Queue Model (LQM) [135, 267] and Double Ring Road Network. The system model may be used to perform analysis of traffic network state properties (e.g., periodicity, stability) and to efficiently compute performance metrics (e.g., timing, flow) for different configurations of traffic networks, control parameters and attack models.

Regarding the differences between this work and aforementioned related transportation system security works: 1) they typically use different metrics (e.g., average waiting time, average queue lengths, or average speed) , 2) they use different traffic network models and dynamics, 3) the system model is uniquely suitable for studying stability properties of states of general networks, and 4) the attack models have a low overhead cost associated with simulations due to the modeling choices [135].

The attack modeling methodology focuses on two control system objectives and one performance metric that an attacker may target: 1) response time, 2) equilibrium system states (NFD), 3) system state stability. One category of attacks isolates the response time as a target and either reduces or increases it according to the stability of the state and its average network flow. Another attack category targets and exploits the state stability itself by modifying the signal settings in a certain manner. Once the signal settings are changed, the system may potentially exit its current state and enter undesirable states. Of primary interest for us is how an attacker may direct the system from a state with high average asymptotic network flow to a state with a lower flow or gridlock behavior (asymptotic zero average network flow).

The methodology does have its shortcomings, including that the traffic model is macroscopic rather than microscopic, that it assumes periodicity for certain states, that it is not able to observe all the possible state-related situations, and that it is not currently able to approximate phenomena such as lane changing. However, despite these shortcomings, the work faces and overcomes several research challenges: 1) determining how, which, and to what extent the controller settings may be attacked, 2) identifying potentially vulnerable system states using periodicity and stability properties from the traffic network models and dynamics, 3) creating attack models that are meaningful (e.g., challenging to detect and/or detrimental to the the average network flow) from both the exploitable control settings and potentially vulnerable states, and 4) simulating and experimenting the attack models and demonstrating their usefulness. While the comparison between the work and other commercial car-following simulation tools, such as SUMO [31], might be helpful, these tools and corresponding car-following models suffer from their own weaknesses (e.g., randomness, complexity) and lack of analytical properties. Thus, this chapter instead aims to serve as a foundation for more rigorous security analysis methodologies for transportation systems and may be compared and integrated with such commercial simulation tools in other works.

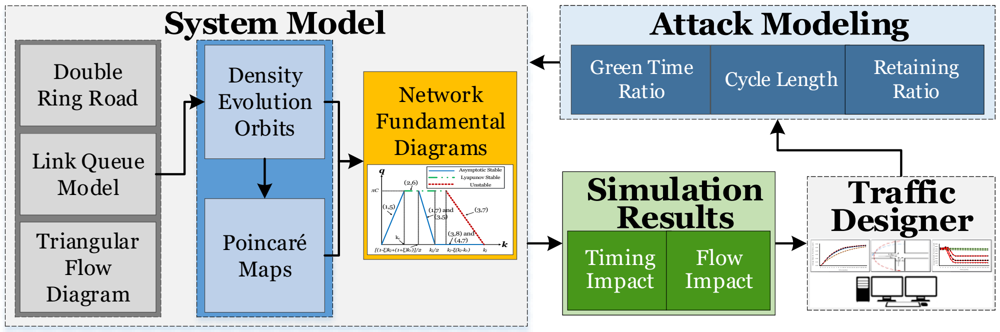


Figure 3.2: The proposed security analysis methodology and its role in the traffic management chain.

The contributions and order of contents of the methodology presented in this chapter are outlined as follows:

1. defining the Double Ring Road and Link Queue Model, and how Density Evolution Orbits, Poincare Maps, and Network Fundamental Diagrams may be derived from them (Section 3.2),
2. providing an overview of attack vectors for fixed-time traffic control systems with wireless

communication capability (Section 3.3),

3. using analytical insights to identify vulnerable states (reducing the state space) and to develop attack models as functions of the traffic network state and control settings (Section 3.4).
4. simulating the attack models and evaluating their impacts in terms of metrics that are unique and "directly related to the decision-making and objectives of traffic control" [200] (Section 3.5).

As shown in Figure 3.2, it is envisioned that this methodology will be useful in the future as part of the traffic management chain to develop (design-time) and maintain (run-time) a provably secure ITS. Where a system is *provably secure* if it is *secure by design/construction* [86]: given an attacker model where the attacker has access to the system and a specified amount of computational resources, the basic security requirements may still be met [154, 98, 225]. In design-time, the methodology may be built upon to choose optimal static control settings for a given traffic network under performance and security expectations. For real-time, efficient strategies may be implemented in response to detection of unexpected attacks to return the system to a state with a more favorable behavior.

3.2 System Model

One may use the Link Queue Model (LQM) to formulate traffic flows in various network levels. This chapter discusses how it may be used in the context of a single ring road network, a double ring road network, and Manhattan-like grid networks.

Background and Related Work

There are several differences between the LQM and models from other works. LQM is able to capture a relationship between the turning ratios, the signal settings, and the initial densities in the form of an NFD derived from stationary/fixed states. Only few other works have considered the turning ratios and signal settings and few works studied multivaluedness (multiple possible flows for a given average density), but even then, these works lack definitions in what potential stationary states are and they lack studies on state stability properties. Works that did recognize the existence of potential stationary states and stability properties lacked in efficiency.

A similar model to the LQM is the Link Transmission Model (LTM) but there are several differences between them. First, the LQM is based on the link's average density, rather than just the number of vehicles over each link. Second, the LQM assumes that the network has an average density that remains constant. And third, the link/road density in the LQM is directly related to supply and demand ordinary differential equations. On the other hand, the LTM uses delayed differential equations as functions of in- and out-fluxes. These delayed differential equations make the LTM infinite-dimensional and not as mathematically tractable as traditional link-based models. Lastly, unlike most other models, the LQM with the Double Ring Road Network Model or grid network can incorporate both signal settings and route choice behaviors and through it, different NFDs for different combinations of them may be derived (whereas NFDs derived from other models only depend on one of or none of these). In addition, before the LQM and analytical work in [89], it was still unclear as to how many stationary states there are in a signalized Double Ring Road Network and what their stability properties are with respect to the signal settings, turning ratios, and initial densities.

Besides the LTM, another related model is the Cell Transmission Model (CTM), which estimates the traffic at each cell in a link rather than the entire link. However, the Cell Transmission Model (CTM) with signalized Double Ring Road used an infinite-dimensional dynamical system due to the kinematic wave model and therefore it is very challenging to solve traffic statics and dynamics. In previous work [135], all these models were qualitatively compared and the comparisons indicate that the LQM is the most efficient in terms of both time and memory. This is because the number of state variables needed to be computed is the number of link densities for LQM; whereas LTM requires two computations and the tracking of older flux values per time step and the CTM requires tracking values per each cell in the link.

The LQM introduced in [135] may be a useful tool for solving freeway and arterial traffic control and observation problems. This is shown by some of its applications published in major research journals such as: Point Queue Models [136], variable speed limit control of a lane-drop bottleneck [133, 137], signalized grid network analysis and control [89], and urban traffic estimation for real-world settings [103].

3.2.1 Single Ring Road and Lab Test-Bed

In a Japanese research experiment where different numbers of drivers drove vehicles within a single ring road, it was observed that there were specific numbers of vehicles that would lead to a change in network behavior within the context of average network flow. These numbers of vehicles are generally denoted as critical densities. For example, as the number of vehicles increased past a critical density, a traffic jam would occur after some time despite no presence of a bottleneck [251]. Within the context of a single ring road model, the LQM also reflects similar behaviors as the Japanese experiment [141]. In order to emulate more realistic traffic behaviors like those in the Japanese experiment, a lab test-bed of robotic vehicles following each other on a single ring road is implemented (Figure 3.3). The robotic vehicles use the

Optimal Velocity Model (OVM) car-following algorithm based on information from their sensors (cameras) and parameters (e.g., permissible distances). Some testing results of the model presented in Table 3.1 matched the behaviors observed in [251]. As the LQM also models this type of behavior, the robotic vehicle experiment was a step above simulation to importantly show how it could model realistic scenarios.

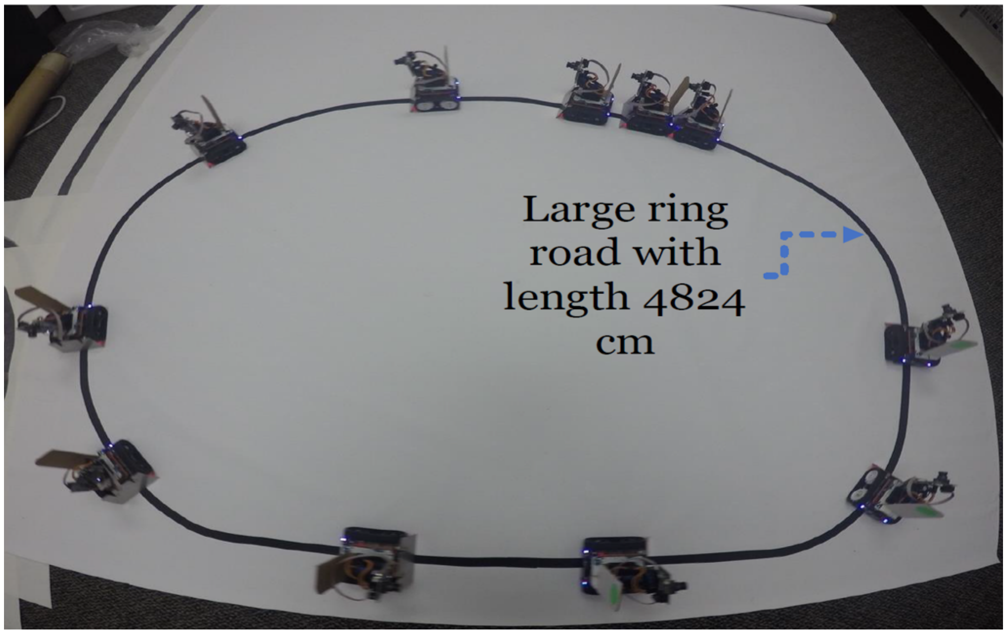


Figure 3.3: A setup of the test-bed in [141] for a large Single Ring Road model with length 4824 cm and 11 robotic vehicles.

Ring Road Size (cm)	**Number of Cars**	**Average Speed** (cm/s)	**Shockwave Speed** (cm/s)
2376	≥ 5	14.5	11
2376	< 5	≥ 20	none
4389	≥ 7	15.5	11.5
4389	< 7	≥ 20	none
4824	≥ 12	21	12
4824	< 12	28	none

Table 3.1: Testing results of single road model testbed.

Unfortunately, the LQM and single road model is not able to capture more interesting behaviors of a traffic network with turning movements. Thus, this work uses a more expanded version of the LQM along with a double ring road network model to cope with such turning movements. To emulate this developed model along with the rest of the attack models in this work, the test-bed is repeatable and may be built upon to do so.

3.2.2 Summary of Traffic Control Settings and System Model Variables

To help the readers follow the chapter, readers may refer to this section for definitions of used traffic control settings and system variables.

- T refers to the traffic signal cycle time (seconds).
- $\pi_i T \in (0,1),\ i \in \{1,2\}$ refers to the effective green time (seconds) where π_i refers to the effective green time ratio for phase i.
- $t_L = T - \sum \pi_i T$ refers to the total lost time t_L (seconds) within the cycle (seconds), which is the sum of yellow and red times, and the time that was not effectively used by vehicles to cross the intersection.
- $\xi_i \in (0,1),\ i \in \{1,2\}$ refers to the retaining ratio of vehicles in ring i.
- L refers to the length of the ring road (miles).
- k_j is equal to 150 vehicles per mile and refers to traffic jam density.
- k refers to the average density in the network (vehicles per mile), $k \in [0, k_j]$.
- k_i^* refers to a fixed system state where the density of ring i will be the same value after a full cycle, $k_i \in [0, k_j]$.

- k_c refers to the critical density and is approximately $\frac{k_j}{5}$ (vehicles per mile).
- q_0 refers to the overall average traffic flow (vehicles per hour).
- R_{p,k_1} refers to the interval that k_1 satisfies for a given DEO region R_p
- $R_{p,k}$ refers to the interval that k satisfies for a given DEO region R_p
- R_p refers to the tuple of the intervals, $(R_{p,k_1}, R_{p,k})$.
- R refers to the set of all R_p, such that $p \in [1, 8]$ as shown in Figure 3.6 (9 and 10 are ignored).
- A DEO is a sequence of visited (k_1, k) regions over a single cycle. DEOs of states are represented as $(p_1, p_2) \mid p_1 \in [1, 4]$ *and* $p_2 \in [5, 8]$.

3.2.3 Double Ring Road Network

The *Double Ring Road Network* model of a *homogeneous symmetric one-way road intersection* is made up of two ring roads that are connected with each other. Larger symmetric one-way road grid networks may also be abstracted with this model (see Figure 3.4). Furthermore, asymmetric and complex networks may also be decomposed into multiple symmetric one-way road grid networks. Additionally, each 2x2 one-way road network is actually similar to a single two-way intersection and an entire traffic network may be converted accordingly. Therefore, this network model and the LQM is applicable for all Urban Traffic Control systems (UTC) [89].

Each ring road has length $L = 0.25$ in miles (mi) and Ring 1 corresponds to South-North roads while Ring 2 corresponds to East-West roads. The connection point of the ring roads helps model turning cars at intersections. This model is based on the concept that the

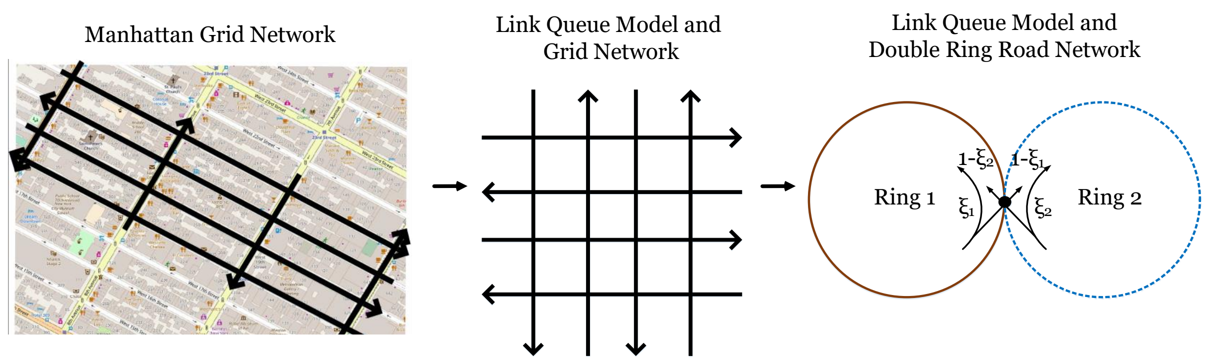

Manhattan Grid Network
Link Queue Model and Grid Network
Link Queue Model and Double Ring Road Network
Ring 1
Ring 2
1-ξ2
1-ξ1
ξ1
ξ2

Figure 3.4: Flow of abstracting an actual grid network (like in Manhattan) to Double Ring Road Network and Link Queue Model.

network will reach a periodic state after a long time, which means that the network's outflow is equal to the network's inflow.

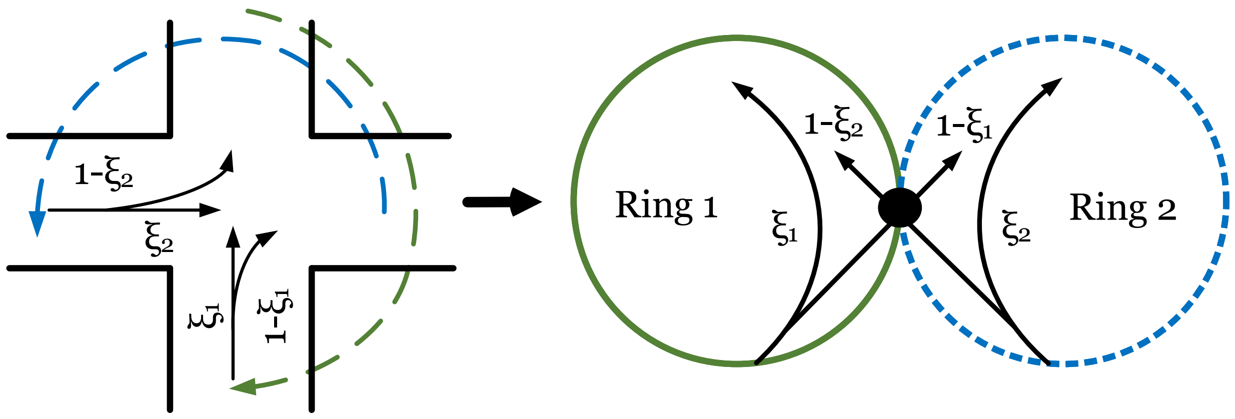


Figure 3.5: A signalized one-way intersection and corresponding abstracted Double Ring Road Network model. ξ_i is the retaining ratio that corresponds to either the N-S or E-W roads in the network model.

In this model, there are two signal phases corresponding to each ring road's turn. Therefore, both the roads and signal phases are identified by $i \in [1, 2]$. In each signal phase, there is an effective green time $\pi_i T$ in seconds, where π_i is the effective green time ratio and T is the total cycle time. The signal regulation is handled by the indicator functions $\delta_1(t)$ and $\delta_2(t)$ in Equations 3.1-3.2. In the first phase, cars from the first ring are moving across the intersection and the cars from the second must wait. The opposite is true for the second phase. In addition to the effective green times, there are also lost times, t_L, per each phase such that $2t_L = T - (\pi_1 + \pi_2)T$. Therefore, it is not required that $\pi_1 + \pi_2 = 1$. The lost time includes the start-up lost time (caused by vehicles' reactions and limited accelerations when signals turn green) and the clearance lost time (caused by wasted yellow and red times).

$$\delta_1(t; T, t_L, \pi_1) = \left\{ \begin{array}{l} 1\text{for } t \in [nT, nT + \pi_1 T], \\ n \in \mathbb{N}_0 \\ 0\text{otherwise} \end{array} \right\} \tag{3.1}$$

$$\delta_2(t; T, t_L, \pi_1) = \left\{ \begin{array}{l} 1\text{for } t \in [nT + t_L + \pi_1 T, \\ (n+1)T - t_L],\ n \in \mathbb{N}_0 \\ 0\text{otherwise} \end{array} \right\} \tag{3.2}$$

Each ring also has a retaining ratio $\xi_i(t)$ and turning ratio $1 - \xi_i(t)$ to specify how many cars remain on the same ring road or turn, respectively. For analytical purposes these retaining ratios are fixed and therefore ξ_i is the notation from now on. $k_i(t)$ is the average density in vehicles per mile (vpm) in ring i over a cycle, and k (vpm) is the total average density in the network. When either or both rings have density equal to the traffic jam density, i.e., $k_i(t) = k_j$, then *gridlock* occurs. Note that, $k_i(t)$ is considered as a variable over time since k is constant. Furthermore, if one knows $k_1(t)$ and k, one may easily derive $k_2(t)$. See Figure 3.5 for a closer visualization of how an intersection is abstracted by the Double Ring Road Network.

3.2.4 Link Queue Model

Behavior Modeling and Formulation

Let S_i in vehicles per hour (vph) denote the amount of supply traffic flow for the downstream link and D_i (vph) denote the amount of demand traffic flow from the upstream link. Both S_i and D_i are constrained by a critical density k_c and are directly related to the traffic influx $f_i(t)$ (vph) and outflux $g_i(t)$ (vph) of the ring roads. The outfluxes are restricted by the demands of the upstream links and the supplies of the downstream links. The definitions for S_i and D_i (Equations 3.3 and 3.4) are derived from the empirical triangular traffic flow diagram in Equation 6 from [89] and are functions of the densities in each ring.

$$D_i(t) = Q(min\{k_i(t), k_c\}) = \left\{ \begin{array}{ll} v_f k_i, & k_i(t) \in [0, k_c] \\ C, & k_i(t) \in [k_c, k_j] \end{array} \right\} \tag{3.3}$$

$$S_i(t) = Q(max\{k_i(t), k_c\}) = \left\{ \begin{array}{ll} C, & k_i(t) \in [0, k_c] \\ \frac{C(k_j - k_i)}{k_j - k_c}, & k_i(t) \in [k_c, k_j] \end{array} \right\} \tag{3.4}$$

where $C = v_f k_c$ is the maximum average flow known as the capacity, v_f is the free flow speed at 60mph, $k_i(t)$ is the average density in ring i over a cycle, and k is the total average density in the network.

Functions for the in-fluxes $f_i(t)$ and out-fluxes $g_i(t)$ may be derived from S and D:

$$g_1(t) = \delta_1(t)min\{D_1(t), \frac{S_1(t)}{\xi_1(t)}, \frac{S_2(t)}{1-\xi_1(t)}\} \tag{3.5a}$$

$$g_2(t) = \delta_2(t)min\{D_2(t), \frac{S_2(t)}{\xi_2(t)}, \frac{S_1(t)}{1-\xi_2(t)}\} \tag{3.5b}$$

$$f_1(t) = g_1(t)\xi_1(t) + g_2(t)(1-\xi_2(t)) \tag{3.5c}$$

$$f_2(t) = g_1(t)(1-\xi_1(t)) + g_2(t)\xi_2(t) \tag{3.5d}$$

The nonlinear ordinary differential equation $\frac{dk_1(t)}{dt}$ may be derived from the above equations (see Equation 3.6). $\frac{dk_1(t)}{dt}$ will be used to compute the evolution of the ring densities over time with respect to the signal settings. Further, from $g_1(t)$ and $g_2(t)$, one may calculate an important metric in traffic engineering: the asymptotic average network flow $q(t)$ (see Equation 3.7). $q(t)$ is bounded by $C = v_f k_c = 900vph$ where the traffic is moving at free flow v_f. Note that it is generally computational heavy to evaluate the integrals of g_1 and g_2.

$$\frac{dk_1(t)}{dt} = \frac{1}{L}(f_1(t) - g_1(t)) = -\frac{(1-\xi_1)}{L}g_1(t) + \frac{(1-\xi_2)}{L}g_2(t) \tag{3.6}$$

$$q(t) = \frac{\int_{s=t-T}^{t} g_1(s)ds + \int_{s=t-T}^{t} g_2(s)ds}{2T} \tag{3.7}$$

3.2.5 System States, Density Evolution Orbits, and Poincare Maps

Using the Double Ring Road Network model and LQM, a system state may be represented by a tuple $(k_1(t), k)$, where $k_1(t)$ is the current density of the first ring road and k the overall average density (from which $k_2(t)$ may be easily derived). A more macroscopic state representation can be made with a Density Evolution Orbit (DEO). A DEO is a sequence (k_1, k) regions visited in a single cycle. It can be attained via analytical insights from the ring road densities and signal settings or derived from simulations. There were 10 unique (k_1, k) regions with unique traffic behaviors, $\frac{dk_1(t)}{dt}$ detected in a previous work ([89]). Regions 1-4 correspond to behaviors in phase 1 ($\delta_1 = 1$) and regions 5-8 correspond to behaviors in phase 2 ($\delta_2 = 1$). Regions 9 and 10 correspond to the times between each phase when no vehicles are moving.

The DEO regions specify just how filled the two rings are and, therefore, the entire network. It is important to know how much density is allocated to each ring at the start because it may correspond to an average asymptotic network flow (if it is a fixed state). The basic idea is that the more symmetric the two ring densities are, the higher the average network flow, while the more asymmetric is it, the lower the average network flow. These regions may therefore serve as a guideline to identify which states are vulnerable and what signal settings must be modified to successfully cause an attack. To identify these regions, for each phase an analysis is performed on the influence of changes in D and S with respect to different relationships between k_1, k_2, k, k_c and k_j on the behavior function $\frac{dk_1(t)}{dt}$. For instance, relationships such as $k_1 > k_c$ and $D_1 = C < \frac{S_1}{\xi_1}$ means that there is a separation between Region 1 and Region 2 at $k_1 = k_c$ because when $k_1 < k_c$, the behavior function will change.

Initial states that only visit one region in each phase, and therefore denote their DEOs as (p_1, p_2) where $p_1 \in [1, 4]$ and $p_2 \in [5, 8]$. In Figure 3.6 depicts an example of a DEO over regions in the (k_1, k) space. Table 3.2 provides bounds and behaviors per each region. From now on, the set of all region boundaries is denoted as R and an individual region as $p \in [1, 10]$ with boundaries for k_1 and k in $R_p \in R$.

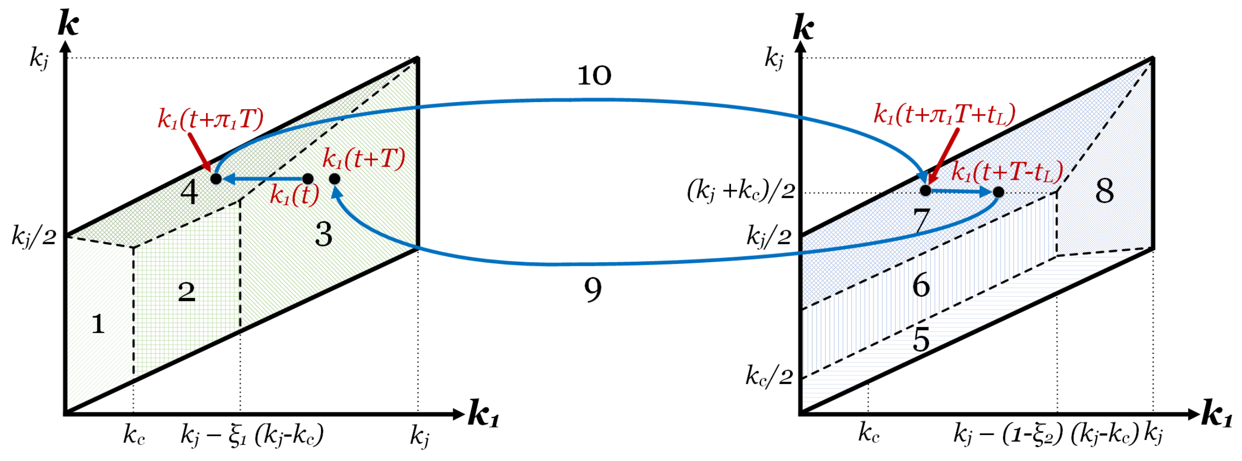


Figure 3.6: Example of Density Evolution Orbit of (k_1, k) over one cycle.

From the initial states, signal settings, and DEOs, one may compute Poincare Maps to analyze traffic network state properties. A Poincare Map [262] is a function P that maps an initial ring density value $k_1(t)$ to a ring density value on the nth cycle $k_1(t + nT)$ and is made up of smaller Poincare Maps for each phase transition. In cases where there is high periodicity such that $k_1(t) = k_1(t + nT)$ for $n \in \mathbb{N}$, $k_1(t)$ is referred to as a fixed state k_1^* with respect to the system signal settings. Potential fixed states where only one region of the (k_1, k) space is visited during each phase were identified in the previous work. These fixed states (k_1^*, k) are of primary focus and are represented by their DEOs.

Region (p)	Signal Phase Indicators $(\delta_1(t), \delta_2(t))$	Conditions $(R_{p,k_1},\, R_{p,k})$	$\frac{dk_1}{dt}$
1	$(1,0)$	$0 < k_1 < k_c,\ \frac{k_1}{2} \le k \le \frac{k_j}{2} - \frac{((1-\xi)k_j-(2-\xi_1)k_c)k_1}{2k_c}$	$\frac{-(1-\xi_1)}{L}D_1$
2	$(1,0)$	$k_c \le k_1 < k_j - \xi_1(k_j-k_c),$ $\frac{k_1}{2} \le k \le \frac{\xi_1 k_j+(1-\xi_1)k_c+k_1}{2}$	$\frac{-(1-\xi_1)}{L}C$
3	$(1,0)$	$k_j - \xi_1(k_j-k_c) \le k_1 \le k_j,$ $\frac{k_1}{2} \le k \le \frac{2\xi_1-1}{2\xi_1}k_j + \frac{k_1}{2\xi_1}$	$\frac{-(1-\xi_1)}{L}\frac{S_1}{\xi_1}$
4	$(1,0)$	$0 < k_1 < k_j,\ max\{\frac{k_j}{2} - \frac{[(1-\xi_1)k_j-(2-\xi_1)k_c]k_1}{2k_c}, \frac{\xi_1 k_j+(1-\xi_1)k_c+k_1}{2}, \frac{2\xi_1-1}{2\xi_1}k_j + \frac{k_1}{2\xi_1}\} < k \le \frac{k_j+k_1}{2}$	$\frac{-(1-\xi_1)}{L}\frac{S_1}{1-\xi_1}$
5	$(0,1)$	$0 \le k_1 \le k_j,\ \frac{k_1}{2} \le k \le min\{\frac{k_1+k_c}{2}, \frac{k_1}{2} + \frac{k_c(k_j-k_1)}{2(1-\xi_2)(k_j-k_c)}\}$	$\frac{(1-\xi_2)}{L}D_2$
6	$(0,1)$	$0 \le k_1 \le k_j-(1-\xi_2)(k_j-k_c),$ $\frac{k_1+k_c}{2} < k \le \frac{k_j+k_1-\xi_2(k_j-k_c)}{2}$	$\frac{(1-\xi_2)}{L}C$
7	$(0,1)$	$0 \le k_1 \le k_j,$ $max\{\frac{k_j+k_1-\xi_2(k_j-k_c)}{2}, \frac{(1-2\xi_2)k_j+k_1}{2(1-\xi_2)}\} < k \le \frac{k_1+k_j}{2}$	$\frac{(1-\xi_2)}{L}\frac{S_2}{\xi_2}$
8	$(0,1)$	$k_j-(1-\xi_2)(k_j-k_c) < k_1 \le k_j,\ \frac{k_1}{2} + \frac{k_c(k_j-k_1)}{2(1-\xi_2)(k_j-k_c)} < k < \frac{(1-2\xi_2)k_j+k_1}{2(1-\xi_2)}$	$\frac{(1-\xi_2)}{L}\frac{S_2}{1-\xi_2}$
9	$(0,0)$	and transition from $(\delta_1(t), \delta_2(t)) = (1,0)$	0
10	$(0,0)$	and transition from $(\delta_1(t), \delta_2(t)) = (0,1)$,	0

Table 3.2: Boundary conditions for region $R_p = R_{p,k_1},\, R_{p,k}$ where $R_p \in R$ that make up a Density Evolution Orbit (DEO)

When the network is in *gridlock*, it means that one or both of the rings is full (k_j). If the network is gridlocked, it is only possible for $(4,7)$ or $(3,8)$ to be the current DEO and they are both fixed for any $\xi \in (0,1)$. The possible DEOs that have fixed states when the **network is not gridlocked are**:

- $(1,5)$, $(1,7)$, $(2,6)$, $(3,5)$, and $(3,7)$ for $0.5 < \xi < 1$;
- $(1,5)$, $(2,6)$, and $(4,8)$ for $0 < \xi < 0.5$;
- $(1,5)$, $(2,6)$, $(4,7)$, and $(3,8)$ for $\xi = 0.5$

With fixed states, the computation of the asymptotic average network flow $q(t)$ can be approximated and redefined with $q(k)$ as $t \to \infty$ instead (see Equation 3.8). This is important as may then map each pair of $k_1^* = k_1(nT),\ n \in \mathbb{N}_0$ and k to an average asymptotic traffic flow q. In turn, this mapping allows us to analytically derive an approximate closed-form formula for a Macroscopic/Network Fundamental Diagram (MFD/NFD), which is useful in practice.

$$q(k;k_1^*) = 2\frac{\int_{s=t-T}^{t} g_1(s)ds}{2T} \approx \frac{g_1(k_1^*) + g_1(k_1(nT + \pi T))}{2} \tag{3.8}$$

Different signal settings are derived for different fixed states (examples are provided in Figure 3.7. The characteristics in the NFDs were consistent with those from previous literature but new ones existed too. Particularly, there were multiple possible flows (i.e., multivaluedness) for some density values and branches with different stability properties. This means that for the same average density, there are two possible values for the average network flow and potential to change signal settings to sway a system from an unfavorable state with low average network flow to a state with higher average network flow. Note, however, that the opposite is of interest for an attacker. Branches represent many possible fixed states, where each fixed state has a stability property.

To analyze the stability properties of fixed states, the network model and Poincare Maps may be combined. It is important to be able to analyze the stability properties of states to understand which ones are undesirable/desirable and critical for proper traffic control. Unstable fixed states are the most vulnerable ones such that even a small perturbation (e.g., from random noise or changes to signal settings) will completely change the system behavior. Lyapunov stable fixed states can handle perturbations up to a certain limit while asymptotic stable fixed states are able to handle any perturbations as long as the signal settings remain the same. Having these properties in mind, potentially vulnerable states are identified and attack impacts on control settings with respect to these states are evaluated.

3.3 Attack Surface

In traffic control networks such as those in Michigan [96], Seattle, New York, and Washington DC [46], wireless technologies are assisting fixed-time traffic control systems by connecting traffic controllers with loop detectors, nearby controllers, traffic management agencies, and vehicles. However, as mentioned, these wireless networks are prone to having or eventually having security vulnerabilities. There are three primary approaches of access (two via exploiting wireless network vulnerabilities) that an attacker can take to modify the signal timing plan. In this section, attack vectors are discussed for how this could be done for each access approach.

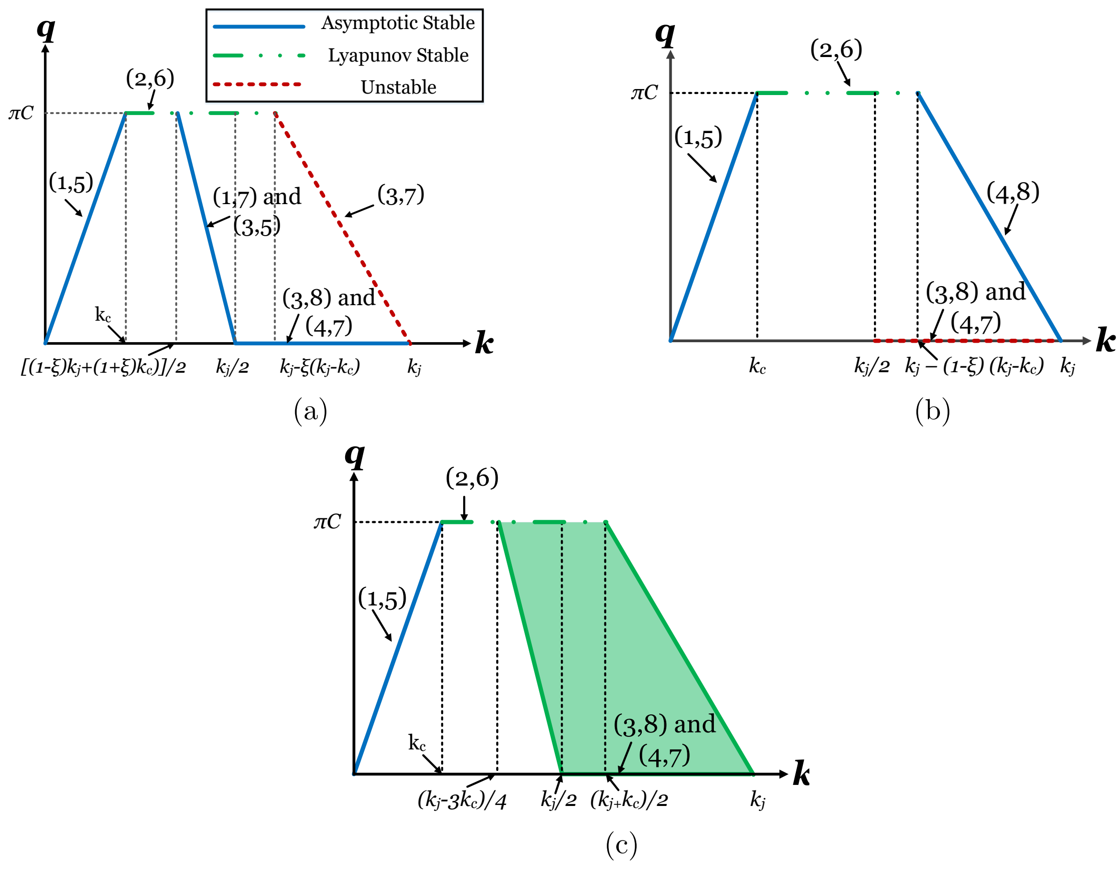


Figure 3.7: Poincare Map-based approximate NFDs when $\pi_1 = \pi_2 = \pi$ for various retaining ratios, $\xi = \xi_1 = \xi_2$. In Figure a, $0.5 < \xi < 1$, in Figure b, $0 < \xi < 0.5$, and in Figure c, $\xi = 0.5$.

3.3.1 Physical/Direct Access

An attacker could open up the traffic controller box (cabinet) to tamper with the equipment and modify any control setting (even remove the fail-safe equipment and cause an all-green light configuration). Obviously, this kind of attack would catch a lot of attention (e.g., video camera detection, suspicious activity reports) and is therefore not particularly a viable method for attackers.

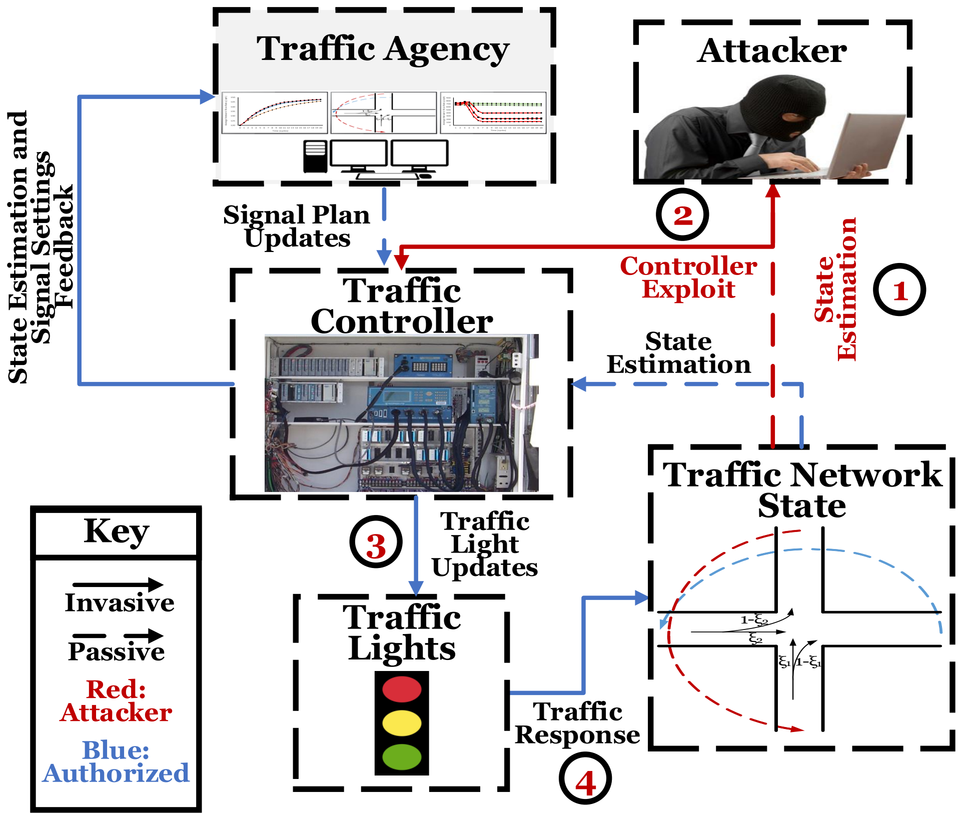


Figure 3.8: Attack Strategy and Attack Vector. **Step 1**) The attacker estimates the system state (may also be done after Step 2), **Step 2**) The attacker performs an exploit to gain controller access (most likely remotely), **Step 3**) The attacker now has authorized access to update the traffic signal cycle at will (under realistic constraints), **Step 4**) The updates will cause one or more detrimental impacts on the average and asymptotic traffic behavior.

3.3.2 Indirect Access

Vehicle detectors and On-Board Units (OBUs) were found to be vulnerable to hacking and could be used for spoofing attacks [55]. Thus, the spoofing attack (e.g., deceiving the controller about existence of cars) would indirectly cause a modification to the signal timing plan at an intersection. Such an attack would have merit for adaptive timing control systems because of the instant impact. For fixed-timing control systems, the attack would have an effect on the decision-making of the traffic management agency and may lead to an update of a less-than-optimal signal timing plan. However, the difference between the attack start time and the response time will reduce the effectiveness of such an attack on fixed-time control systems.

3.3.3 Remote Access

This section will detail how an attacker may directly modify the timing plan via wireless communication. In order to perform a remote access attack to the traffic controller, an attacker must first obtain access into the wireless network of the traffic control system. Cerrudo (2014) and hena et al. (2014) observed through their experiments that there is a tendency for poor or nonexistent security in traffic controller wireless networks. This is because of either the carelessness or insufficient knowledge of controller installers and manufacturers. Although vendors of these vulnerable controllers were kept private in these works, Cerrudo (2014) studied controllers that were distributed in USA and 10 other countries while Ghena et al. (2014) studied controllers from a different vendor deployed in Michigan.

Let us take the controllers that Ghena et al. (2014) studied as an example on how an attacker may gain access. First, an attacker would need a radio wireless card matching the same frequency as that of the controller (in this case, 5.8GHz or 900MHz). Then, they would need to implement the same network protocol, which they can discover via social engineering attack or a reverse engineering attack (slightly more complicated if frequency hopping is implemented, but still feasible), and gain access to the private network by exploiting the security vulnerabilities (e.g., weak encryption keys, passwords). A traffic controller network for a specific intersection could then be accessed more than half a mile away. They could also attempt to use a drone to perform a mobile attack. Either way, the connectivity range is attractive for attackers who wish to remain undetected throughout the lifetime of the attack [95].

Modifying Traffic Signal Settings

Having access into the private controller network implies access to all other controllers on the same network. However, even if an attacker had access to a traffic control system, *due to hardware-based solutions (malfunction management units) they cannot force unsafe signal combinations (e.g., green-green or red-red)* [96, 269]. Nevertheless, the attackers would still be able to modify the scheduling of traffic signals or force a blinking red phase (but the latter is easily detectable).

Continuing with the previous example, modifying the traffic signal settings could be done via two methods after gaining network access: 1) sending memory modification commands to the debug port in the controller's VxWorks OS or 2) using remote control commands provided in the National Transportation Communications for ITS Protocol (NCTIP) 1202. The VxWorks debug port issue may have been patched, but the remote control attack vector remains. Remote control commands include malicious logic statements, activating any button on the controller, or modifying light timings (shorten or lengthen phase times). Since all the controllers are connected in a one-hop manner to their neighbors, an attacker may perform a small or large scale attack depending on their resources and objectives.

Attaining System State Knowledge

With or without access to the wireless network, the attacker may easily attain knowledge about the current state of the system, which includes the current timing plan configuration and the physical state of the intersection (i.e., densities and DEO). If they had no access to the network, the attacker can easily use their own sensors (e.g., phone, camera) or observations because the traffic intersection is in a public space. While if they did have access, they can use the measurements from existing loop detectors and/or cameras. One may also assume that, with the introduction of more technology (i.e., image processing, smarter sensors) it will be even easier to accurately estimate the current system state.

Attack Timing

Despite the different possible roles, it is assumed that the attacker does not desire to be easily detected. This means that their attacks may be subtle yet may have profound long-term effects on the system behavior. Fixed timing signal plans are regularly updated (even hourly) to address traffic demands or for reasons such as major traffic changes, time of day, and inspections. If the attacker can modify the signal timings around the same time that the timing plan is regularly updated or quickly reverse their modifications after a short time, the changes will not be trivially detectable. Even if detected, it may be too late (response time varies considerably [269]), as the impact from an attack may already have forced the system en route to a targeted state, despite reversed settings by the agency.

After gaining some level of access the attacker can perform an attack on the integrity (i.e., the signal timing) and consequently the availability (i.e., traffic flow, control response time) of the traffic control system. For a graphic overview on the steps an attacker would take discussed in this section, please refer to Figure 3.8. Note that it is general enough to consider the different potential controller exploits discussed in this section. To assess the impacts of such attacks, attack models are defined, potentially vulnerable states are identified, and attack impacts are evaluated.

3.4 Attack Modeling & Impact Analysis

An attack model consists of the system model (LQM simplified as first DEOs and then Poincare Maps for analytical purposes) and attacks on various system control parameters through vulnerability exploits in the aforementioned attack surface. In a sense, an attack model is similar to a closed-loop control system model but where the attacks are anti-controls. In this section, by using the Density Evolution Orbits (DEOs) and Network Fundamental Diagrams (NFDs) in Section 3.2.5, potential initial and final (targeted) state tuples derive average network flow attack impacts from the Poincare Map-based NFDs that are most desirable for an attacker. Then, along with the behavior differential equations in Section 3.2.4 and limitations of the attacker in Section 3.3, attacks on the controller settings to achieve these impacts are devised.

The general attack model is defined as: $AM(M, x_0, \Delta) =$

$$\left\{\begin{array}{l} \text{Convergence Time Impact: } t_{conv,\Delta},\ (t_{conv,\Delta} - t_{conv}), \\ \text{for } DEO_\Delta = DEO_0 \text{ and } \pi_{1,\Delta} = \pi_{2,\Delta} \\ \text{Asymptotic Average Flow Impact: } q_\Delta,\ (q_\Delta - q_0) \\ \text{for } DEO_\Delta \neq DEO_0 \text{ and } \pi_{1,\Delta} \neq \pi_{2,\Delta} \end{array}\right\}$$

First, inputs are defined for the AM. The system model M is a vector of the following: NW, $k_i(t)$, k, π_i, $\delta_i(t)$, $\xi_i(t)$, $D_i(t)$, $S_i(t)$, $f_i(t)$, $g_i(t) \mid i \in \{1, 2\})$ where the network variable NW is a tuple with the number of intersections with link length L and the initial cycle length T (see Section 3.2 for definitions of the other variables). The initial system state x_0 can be defined in terms of the original traffic control settings and initial traffic densities of each road before the start of the attack. The traffic control settings at time t_0 include π_1, π_2, and T. Traffic densities are $k_1(t_0)$, $k_2(t_0)$, and k.

An intelligent attacker is considered, whose objective is to force the system into a targeted density evolution behavior via modifying control parameters. The variable Δ is a tuple of the modifications and their timings that the attacker plans to perform: the new cycle length, T_Δ; the new effective green time ratios $\pi_{1,\Delta}$ and $\pi_{2,\Delta}$; the starting time of the attack t_{start}, the duration of the attack t_Δ and ending time of the attack $t_{start} + t_\Delta$ (when the modifications return to normal), and the ending time of the simulation/measurements t_{finish} s.t. $t_{finish} \geq t_{start} + t_{\Delta}$; the starting state of attack x_0 with DEO_0 and time t_0, and ending state of attack x_{Δ} with DEO_{Δ} at time t_{finish}.

In this chapter, it is assumed that only one type of signal setting (green time ratios, cycle length) will have nonzero modifications during an attack. Modifying the allocations of green time or modifying the cycle length itself will directly affect the effective green time ratios. Modifications on these settings are only studied because, despite having the ability to modify multiple other parameters, an attacker would be extremely interested in a low number and amount of modifications to destabilize the system. Furthermore, modifications to the retaining ratios are not considered (which could be changed through route guidance application exploits) in this chapter. Although combinations of parameter modifications are not considered in these attacks, the attack strategies discussed in this work are simple yet may cause serious impacts on the traffic network. However, one may argue that an attacker would be most interested in the simplest changes to create a sufficient amount of havoc. Additionally, this work is a foundational basis to other works that tackle research problems such as: deriving the most effective attack or analyzing and understanding the effects of modifying several different parameters at the same time.

There are two types of attack behavior categories for AM that will be studied in this section. The first category - Non-State-Changing - speeds up or slows down the system's convergence to a fixed state behavior (especially the gridlock state). The second category - State-Changing - causes one or several state change(s) and reduces the asymptotic average flow. For Non-State-Changing attacks, the new time at which the system will converge to its expected stationary behavior is denoted as $t_{conv,\Delta}$ and the original convergence time is t_{conv}. Thus, the impact is $t_{conv,\Delta}$ - t_{conv}, where t_{conv} depends on the designer-based range of permissible performance metrics (e.g., densities, asymptotic average flow-rate). For both attack categories, in particular State-Changing attacks, another impact is difference in average flow rates, $q_{\Delta} - q_0$, which can be predicted from the initial and final DEOs (if known) for states, x_0 and x_{Δ}.

3.4.1 Non-State-Changing Attacks

Non-state-changing attacks will not theoretically change the DEOs of fixed states if the attacker chooses to modify the cycle length or the green time ratios such that $\pi_{1,\Delta} = \pi_{2,\Delta} = \pi_{\Delta}$. Instead these modifications will cause the expected asymptotic or stationary behavior of the state to be reached at a different time $t_{conv,\Delta}$ instead of the original expected time t_{conv}. Original settings are assumed to be $\pi_1 = \pi_2 = \pi \neq \pi_{\Delta}$. In the following sections, attacks are considered to speed up the convergence of asymptotic stable gridlock states. Recall from Figure 3.7 that asymptotic stable gridlock states are states with DEO of either (3, 8) or (4, 7). Thus, a designer and an attacker would be strongly interested in identifying when the system is in a state with one of these DEOs.

Gridlock Speed-Up Attack When $DEO_0 = DEO_{\Delta}$ = (3, 8)

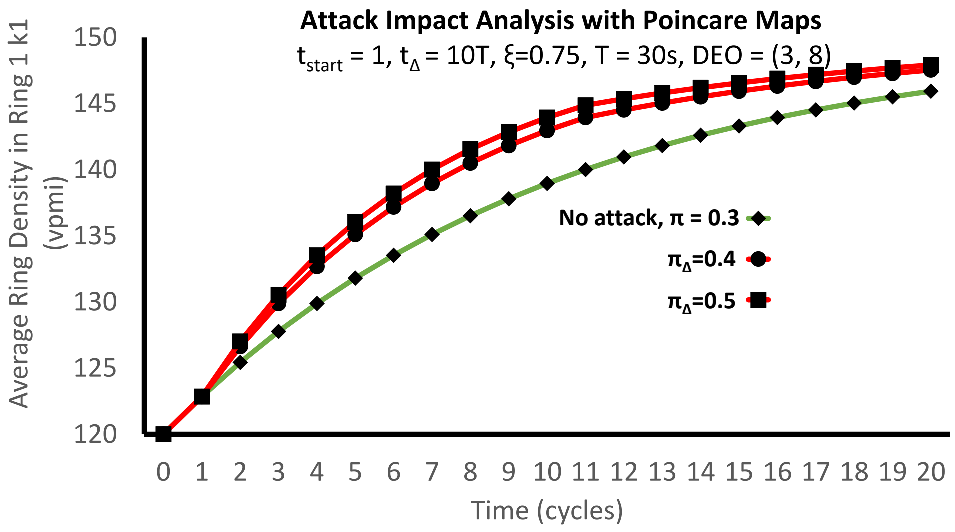
Given $\pi = \pi_1 = \pi_2$, $\xi > 0.5$, $k_1(0), k$ and $DEO_0 = DEO_{\Delta} = (3,\ 8)$, an attack with π_{Δ} such that $\pi_{\Delta} = \pi_{1,\Delta} = \pi_{2,\Delta} > \pi$ will create a new convergence time $t_{\Delta,conv}$ such that $t_{\Delta,conv} < t_{conv}$. For a DEO of (3, 8), that eventually k_1 will reach k_j given the signal settings from the previous work. The Poincare Map equation corresponding to this DEO is $k_1(nT) = k_j(1 - e^{(\gamma_2 - \gamma_3)\pi nT}) + k_1(t)e^{(\gamma_2-\gamma_3)\pi nT}$ where $\gamma_2 = ((1-\xi_1)k_c)/L\xi_1(k_j - k_c)$ and $\gamma_3 = v_f k_c / L(k_j - k_C)$. Since the exponent includes the green time ratio and since $\lambda > 0$, then it is easy to see that with values of π_{Δ} such that $\pi_{\Delta} > \pi$, the rate of density growth will increase and therefore lead to $k_{1,\Delta}(t_{start} + t_{\Delta}) > k_1(t_{start} + t_{\Delta})$ where $k_{1,\Delta}$ is the ring road density when there is an attack while k_1 is the normal expected density without attack. Because of this, even if the growth rate returns to normal after attack completion, the system will still more quickly converge to the limit, i.e., $t_{conv,\Delta} < t_{conv}$. Using the same logic, similar impacts will occur when the attacker modifies the cycle length such that $T_\Delta > T$.

In Figure 3.9, convergence time impacts of this attack model are demonstrated for different π_Δ where $t_{Start} = 1T$ and $t_\Delta = 10T$. Poincare Map settings include: $L = .25$, $\pi = 0.3$, $\xi = .75$, $k_1(0) = 120$, and $k = 75$. When $T = 90s$, the maximum impact logically occurs for $\pi_\Delta = 0.5$ and is $t_{conv} - t_{conv,\Delta} = 20T - 13T = 7T$. Hence, there is a gridlock convergence speed up of $20T/13T = 1.53$. This means that the agency has 7T less time to detect and respond with setting changes to guide the system to a favorable state instead.

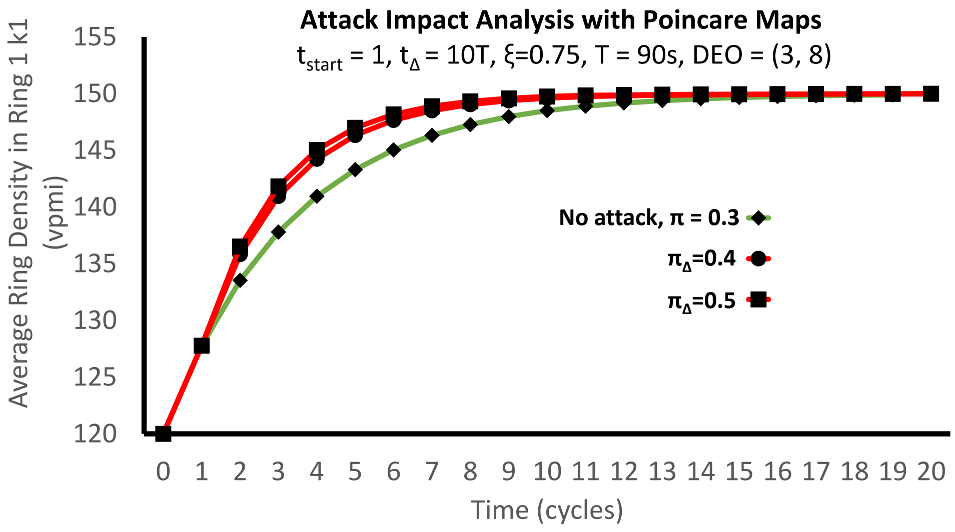
Although it appears that the system reaches the steady state value around nearly the same time, the biggest concern should be that even just a few cycles of reduced convergence time may be highly costly for drivers and the traffic agency. This is because the later the reaction of the traffic control system authorities, the more pronounced the effects of the attack and the more difficult it is to reverse the attack. In addition to looking at just the asymptotic limit, it is important to note the trajectory of traffic behavior from start to finish and how there are pronounced effects before the asymptotic limit is reached. Furthermore, these figures describe an example where the attack was for a finite duration (10 cycles) and not from start to finish. If the attack was performed the entire time, it would be clear that the convergence would be even more rapid.

Gridlock Speed-Up Attack When $DEO_0 = DEO_\Delta =$ (4, 7)

For $DEO_0 = DEO_\Delta = (4,\ 7)$ when $\xi > 0.5$, eventually k_2 will reach k_j and therefore k_1 will eventually reach $2k - k_j$. Similar to the previous case, the Poincare Map equation for an initial state with this DEO can be used to compute how much more quickly the system reaches gridlock, $t_{conv,\Delta} \leq t_{conv}$, when $\pi_\Delta > \pi$ or $T_\Delta > T$.



(a) $T = 30s$ and attack timing settings are $t_{start} = 1T$ and $t_{\Delta} = 10T$.



(b) $T = 90s$ and attack timing settings are $t_{start} = 1T$ and $t_{\Delta} = 10T$.

Figure 3.9: Graphs of Poincare Maps for k_1 with $\pi = 0.3$, $\xi = 0.75$ and initial states with DEO = (3, 8) where $\pi_{1,\Delta} = \pi_{2,\Delta} \in \{0.4, 0.5\}$

3.4.2 State-Changing Attacks

For state-changing attacks, the modified green time ratios will satisfy $\pi_{1,\Delta} \neq \pi_{2,\Delta}$. When $\pi_{1,\Delta}$ and $\pi_{2,\Delta}$ are unequal, it is difficult to predict the density growth and intermediary states may be non-fixed states. Therefore, analytical or numerical solutions may not only be relied upon and the model should be simulated to prove and define an attack impact on the asymptotic average network flow. However, one may still reduce the complexity of attack impact estimation for state-changing attacks. This may be achieved by identifying states with DEO_0 that are vulnerable to attacks with target DEO_Δ through an "estimation" of the direction of the growth of (k_1, k) from the sum of the exponential coefficients in the Poincare Map equations. Let λ in Equation 3.9 be the sum of these exponential coefficients. If λ is zero, then the system is in a possibly fixed state. If λ is nonzero, the density evolution is nonzero and k_1 is either constantly increasing or decreasing.

$$\lambda(k_1, k) = A_{p_2}k_1 + B_{p_2} + A_{p_1}k_1 + B_{p_1} \tag{3.9}$$

where $p_1 \in [1, 4]$ & $p_2 \in [5, 8]$ and A and B are exponential coefficients corresponding to the region-based definitions of the differential equation $\frac{dk_1}{dt}$ (which has been derived and defined in previous work)

Given the attack model, one may identify if an initial state is vulnerable if the direction of density evolution leans toward the regions of the targeted DEO, DEO_Δ. Therefore, given an initial state x_0 with a DEO_0, the direction of growth is studied under malicious modifications with an updated version of λ denoted as λ_Δ in Equation 3.10. Note that an attack on one type of parameters is only considered (either both green time ratios or the cycle length).

$$\lambda(k_1, k)_\Delta = (A_{p_2}k_1 + B_{p_2})\pi_{2,\Delta}T_\Delta + (A_{p_1}k_1 + B_{p_1})\pi_{1,\Delta}T_\Delta \tag{3.10}$$

where $p_1 \in [1, 4]$ & $p_2 \in [5, 8]$

Given DEO_0 and $\pi_{1,\Delta}$ and $\pi_{2,\Delta}$, all region boundaries R, Poincare Maps, and NFDs for $\xi = 0.5$ and $\pi_1 = \pi_2 = \pi$ (see Figure 3.7), λ_Δ may be used to discover potentially vulnerable states with DEOs of interest to an attacker. As an example, two types of initial DEOs of interest for an attacker are discovered. These DEOs are (2, 6) and (4, 8). These initial DEOs correspond to "possible stationary" states according to the Poincare-Map based NFDs. States with DEO of (2, 6) have k_1 and k values that satisfy R_3 and R_8 when $\xi = 0.5$ but not when $\xi \neq 0.5$. Similarly, initial states with $DEO_0 =$ (4, 8) share common ranges of k_1 and k values with regions R_3, R_8, R_4 and R_7. Hence, for demonstration purposes, states that are potentially vulnerable to State-Changing attacks are described: 1) $DEO_0 =$ (2, 6) and $DEO_\Delta =$ (3, 8), and 2) $DEO_0 =$ (4, 8) and $DEO_\Delta \in \{(4, 7), (3, 8)\}$. For all potentially vulnerable initial states, one may analytically estimate and infer some insights from the attack impact $q_\Delta - q_0$ from the initial (DEO_0) and targeted (DEO_Δ) DEOs, as $q(k)_{DEO_0} - q(k)_{DEO_\Delta}$ using the NFDs. However, since the attack impact is also a function of k_1, simulation is required to compute the actual k_1 at the end of the attack at time $t_{start} + t_\Delta$.

Scenario When $DEO_0 =$ (2, 6) and $DEO_\Delta =$ (3, 8)

Given $x_0 = (k_1(0), k)$ with DEO_0 of (2, 6) and $\xi_1 = \xi_2 = \xi = 0.5$. From the definitions of the region boundaries, it is clear that k_1 must increase for a change in DEO from (2, 6) to (3, 8) as long as k satisfies the boundaries of k for all regions R_2, R_3, R_6, R_8. For this scenario to be successful, $\lambda_\Delta > 0$ for the DEO of $x(t)$ must be satisfied so that (k_1, k) will increase and move toward a state with DEO of (3, 8) according to the region boundary definitions in the analytical studies. In the initial state with $DEO_0 =$ (2, 6), it is notable that $\lambda = 0$ since it is considered a stationary Lyapunov stable state. Thus, if $\pi_{1,\Delta} < \pi_{2,\Delta}$ then λ_Δ will be greater than zero.

Scenario When $DEO_0 = (4, 8)$ and $DEO_\Delta \in \{(4,7),(3,8)\}$

When the initial DEO is (4, 8) and $k_1 = k$ for stability and k satisfies the boundaries of (4, 7) or (3, 8), an attack may be successful as long as $\pi_{1,\Delta} > \pi_{2,\Delta}$ and $\xi \leq 0.5$ for $DEO_\Delta = (4,7)$ or $\pi_{1,\Delta} < \pi_{2,\Delta}$ for $DEO_\Delta = (3,8)$. This idea can also be extracted from similar logic to that in the previous section.

3.5 Experimental Results

The validity of each attack model will be confirmed by simulation with different levels of complexity.[1] First, LQM and the Double Ring Road Network model are used. Then, going up a level of complexity, the impacts of the same attacks on a symmetric one-way grid network model of multiple intersections are compared. For the experiments, $T = 30s$ and $T = 90s$ for various runs, and $L = .25$ and $t_{finish} = 20T$. The update time step is set to 0.05s. Table 3.4 contains information on the range and number of initial states tested for each attack model. $DEO_0 \in [(3,\ 8),\ (4,\ 7)]$ correspond to Non-State-Changing attacks and $DEO_0 \in [(2,\ 6),\ (4,\ 8)]$ correspond to State-Changing attacks.

3.5.1 LQM with Double Ring Road Network

Non-State-Changing Attacks on Asymptotic Gridlock States

To validate the analytical observations in Section 3.4.1, the attacks are simulated with the LQM and Double Ring Road Network. Below are figures describing the impacts of different modified green time ratios on the asymptotic stable gridlock states under the constraint that $\pi_{1,\Delta} = \pi_{2,\Delta}$ and the DEO does not change. The following two cases where $\pi_\Delta = \pi_{1,\Delta} =$

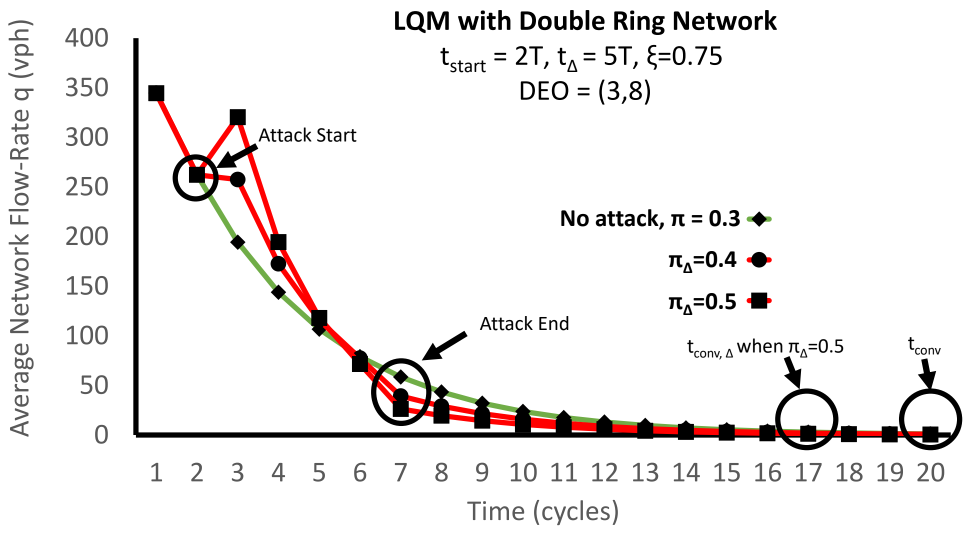
[1]All simulation code for this section is provided in *https://github.com/AICPS/LQM_traffic_sec_official*.

$\pi_{2,\Delta} \in \{0.4, 0.5\}$ are considered: 1) $t_{start} = 2T$ and $t_{start} + t_\Delta = 7T$ (Figures 3.10a-3.10b), 2) $t_{start} = 1T$ and $t_{start} + t_\Delta = 11T$ (Figures 3.10c-3.10d).

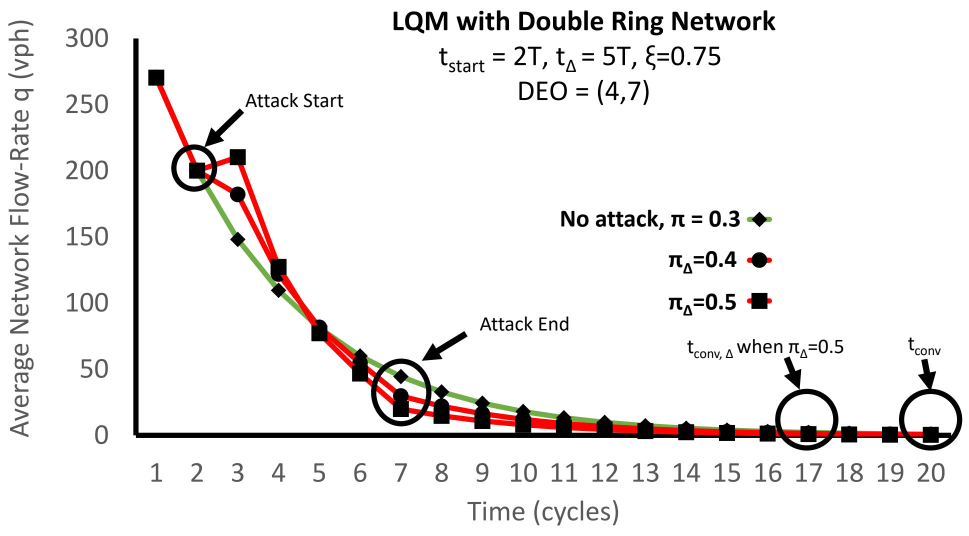
As can be seen in Figures 3.10a-3.10b, for an attack with $t_{start} = 2T$ and $t_\Delta = 5T$ with $\pi_\Delta = 0.5$, the approximate average impact is $t_{conv,\Delta} - t_{conv} = 20T - 17T = -3T$ (1.17 convergence speedup) for both DEOs. In Figures 3.10c-3.10d, $t_{start} = 1T$ and $t_\Delta = 10T$. When $\pi_\Delta = 0.5$, the impact is $-6T$ (1.4 convergence speedup) for initial states with DEO = (4, 7) and $-7T$ (1.5 convergence speedup) for initial states with DEO = (3, 8). The sudden yet temporary rise in flow for attacks on initial states with DEO = (3, 8) is possibly due to order of phases (ring 1 first) and the increase in green time. As this is just a motivational case study, more severe impacts may occur from longer-lasting or earlier-starting attacks. Additionally, combinations of different attack categories may add up to more detrimental impacts.

Having an effective green time ratio of 0.3 may be deemed as unrealistic in practice since this would mean for a cycle length T=90s, the lost time tlost would be 18s and each effective green time would be 27s. Although this satisfies the minimum green time and maximum green time constraints for a traffic signal cycle of 90s, this would be quite unrealistic in practice since a lot of time would be wasted for just waiting. Therefore, a closer-to-realistic case of 13.5s for each lost time is considered where the effective green time ratios would then be 0.35 for both phases and 9s for each lost time for effective green time ratios of 0.4. Thus, when the attacker gains access, when the system is in an asymptotic gridlock state with DEO = (3, 8) or (4, 7), they have the option of increasing the cycle length or decreasing the lost time to accelerate the convergence to gridlock (since the effective green time ratios are increased as a result). However, they are still constrained by the minimum lost time, which is about 3s per phase. Thus, the most the attacker can force the effective green time ratios for T=90s would be about 0.48 where 43s would be the green time for each phase.

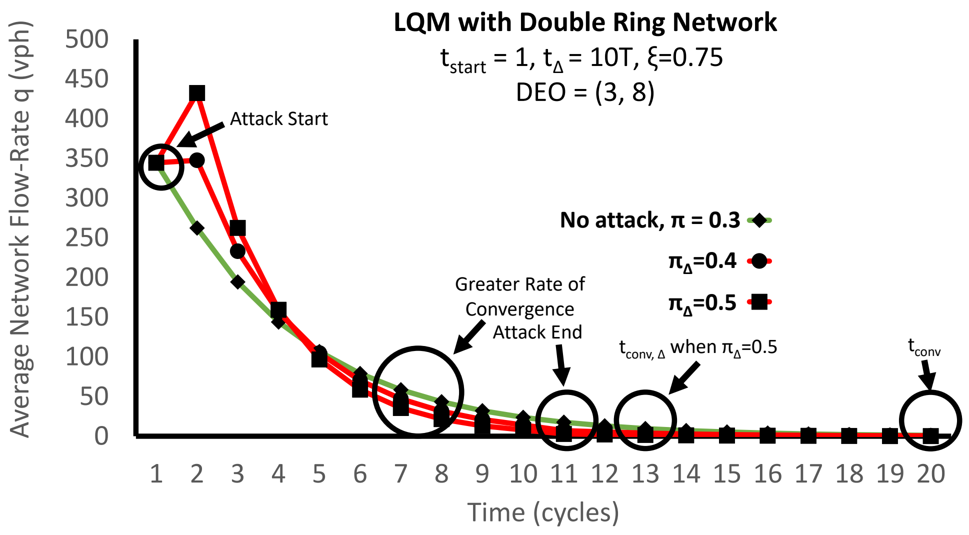
In Figure 3.11, the simulation results for this situation for $\xi = 0.75$, π=0.35, $\pi_\Delta = 0.48$,



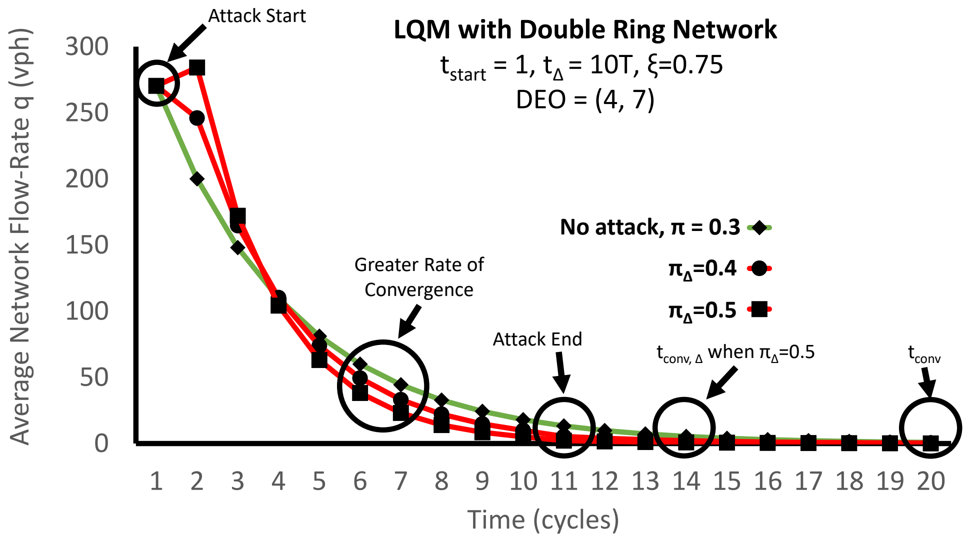
(a) $DEO_{\Delta} = (3, 8)$, $T = 30s$, $t_{start} = 2T$, and $t_{\Delta} = 5T$.



(b) $DEO_{\Delta} = (4, 7)$, $T = 30s$, $t_{start} = 2T$, and $t_{\Delta} = 5T$.



(c) $DEO_\Delta = (3, 8)$, $T = 90s$, $t_{start} = 1T$, and $t_\Delta = 10T$.



(d) $DEO_\Delta = (4, 7)$, $T = 90s$, $t_{start} = 1T$, and $t_\Delta = 10T$.

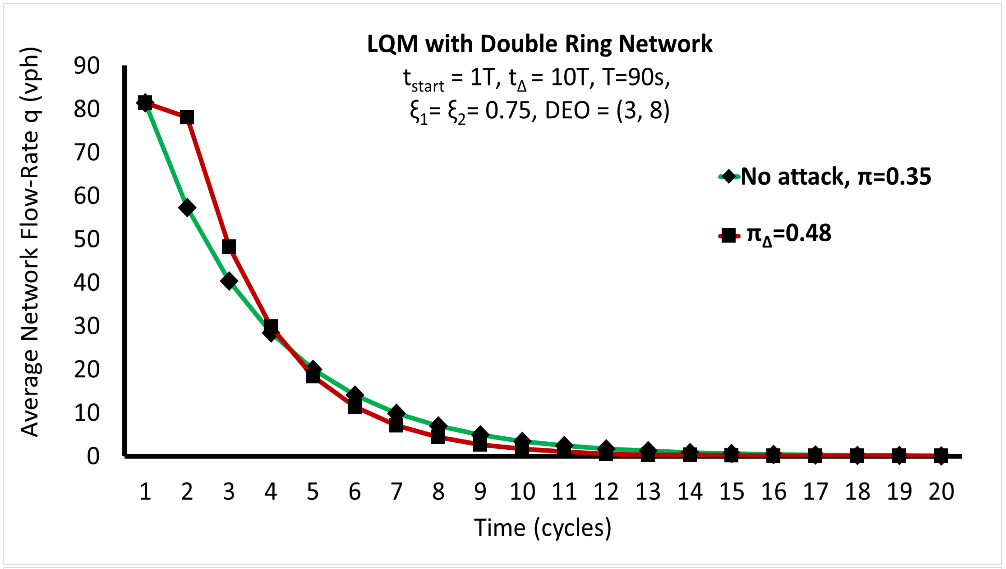
Figure 3.10: Non-state-changing attack simulations for LQM and Double Ring Road Network with $t_{finish} = 20T$, $\pi = 0.3$, $\xi = 0.75$, $\pi_\Delta = \pi_{1,\Delta} = \pi_{2,\Delta} \in \{0.4, 0.5\}$.

$t_{start} = 1T$ and $t_\Delta = 10T$, thus showing that the attack is very capable of reducing the control response time for the system engineers. One may see that the line corresponding to no attack hits very close to gridlock (less than 0.5 average flow-rate) at around 16T and the attack line hits very close to gridlock at 11T in the figure. This means the convergence speed-up is 16T/11T, which is approximately 1.5x. If the initial effective green time ratios is set to 0.4, the initial convergence time is around 15T and the behavior is quite close to when they are initially 0.35.

Although demonstrated and studied in previous work, some results are provided for when the retaining ratios are chosen randomly from a range of values under the studied non-state changing attack model. For the non-state-changing attack model, a retaining ratio of 0.75 is assumed in the simulations. In the following results, a random retaining ratio is chosen from the range of [0.51-0.99] for each ring (common in practice and in reality as well [224]). In Figures 3.12 and 3.13, the effects of having asymmetric and random retaining ratios for each ring ($\xi_1 = 0.901$, $\xi_2 = 0.813$ and $\xi_1 = 0.571$, $\xi_2 = 0.945$) are observed on the system and attack modeling behaviors. It may be observed that the overall behaviors are similar to the results in the previous graphs despite the random and unequal retaining ratios. Note, there are some clear unique behaviors for the simulations regarding $\xi_1 = 0.571$ and $\xi_2 = 0.945$ for $DEO_\Delta = (3, 8)$ because the difference between ξ_1 and ξ_2 is large and the larger retaining ratio is for the less congested ring (ring 2). Thus it will take a longer time to converge to the asymptotic limit. Results for other identified vulnerable states provide similar insights and confirm that as long as the retaining ratios are within the specified range of [0.51-0.99], the attack model is applicable.

State-Changing Attacks on Initial Stationary States

In Figure 3.14a, the initial states have a DEO of (2, 6) and the initial settings are $\pi = \pi_1 = \pi_2 = 0.5$, $T = 90s$. The attack parameters include $(\pi_{1,\Delta}, \pi_{2,\Delta}) \in \{(0.4, 0.6), (0.35, 0.65), (0.3, 0.7)\}$



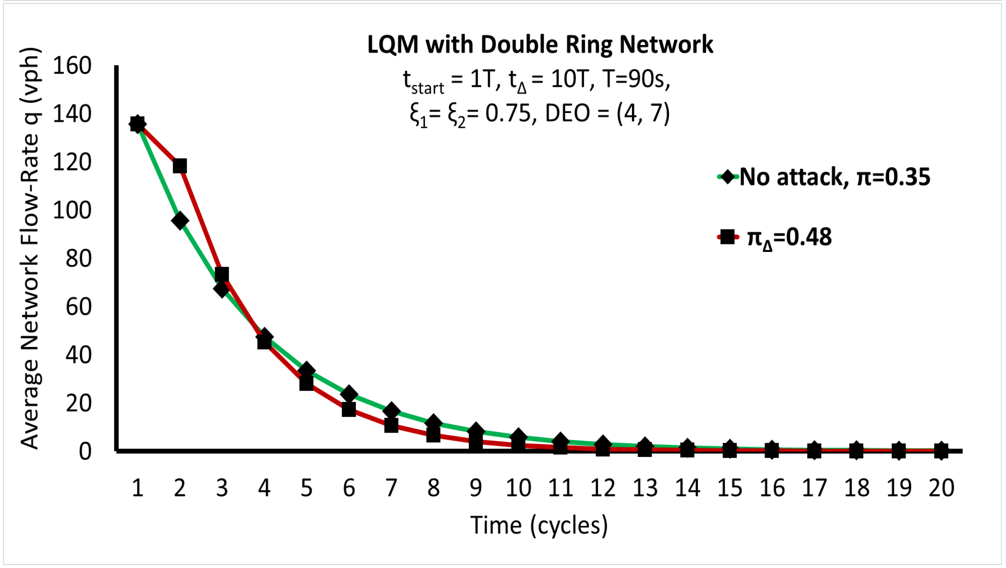
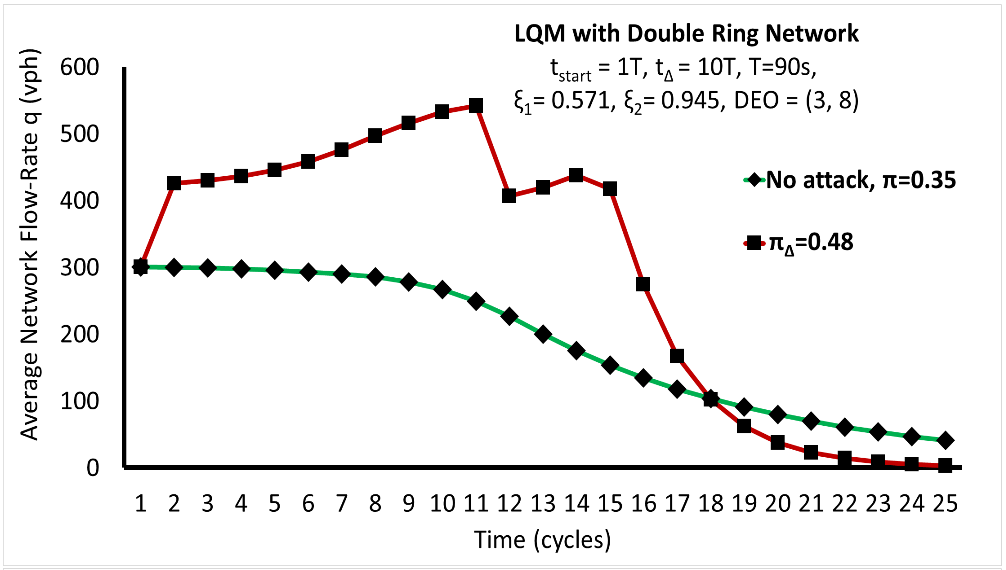


Figure 3.11: Non-State-Changing Attack Model Simulations with higher and more realistic initial effective green time ratio π=0.35 and modified effective green time ratio π_Δ=0.48. These effective green time ratios are more realistic because the lost times are more reasonable per cycle. Acceleration of convergence time and reduction of control response time is apparent here just as it is in previous simulations.



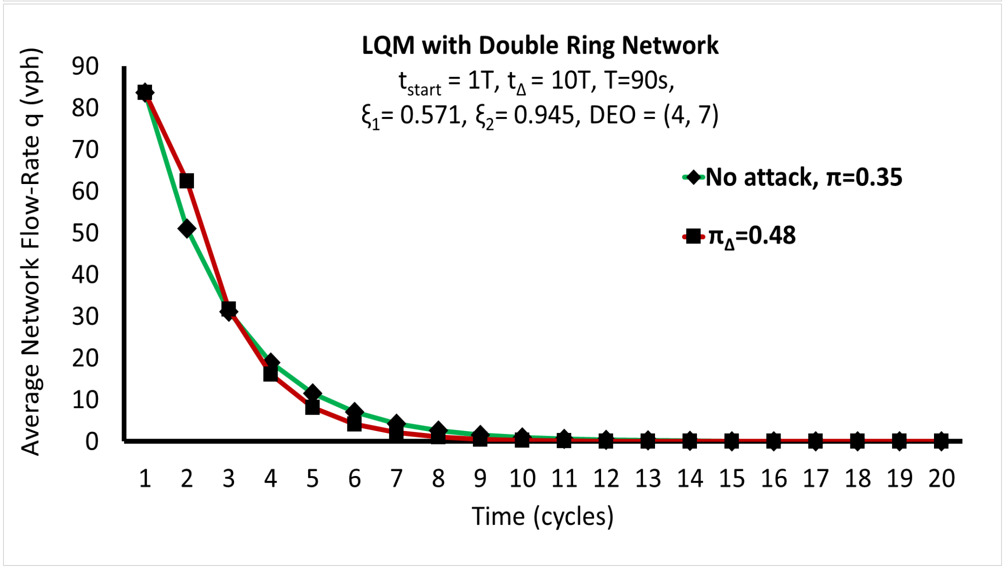
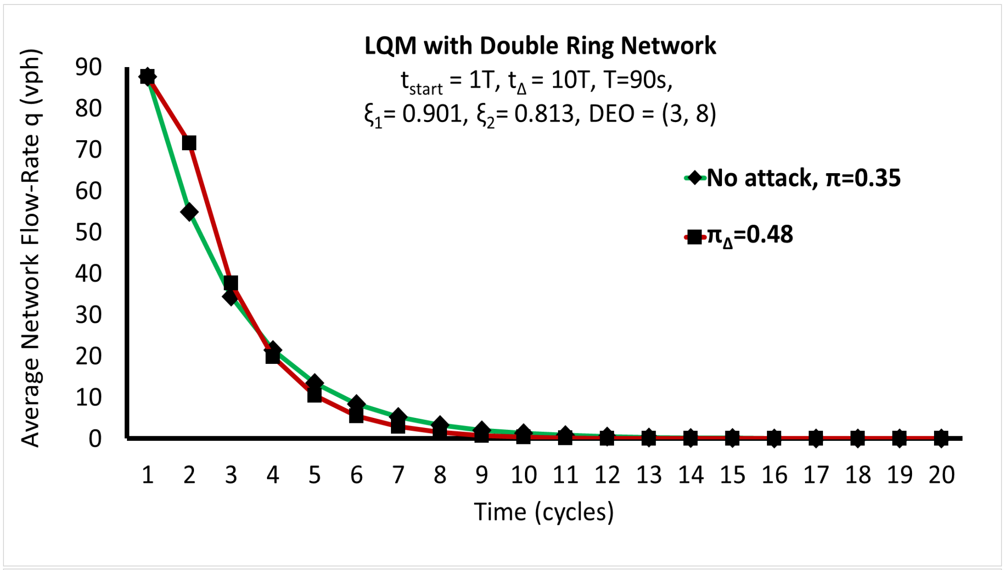


Figure 3.12: Non-State-Changing Attack Model Simulations with Randomly Selected and Asymmetric Retaining Ratios, $\xi_1 = 0.571$ and $\xi_2 = 0.945$. For a targeted $DEO_{\Delta} = (3, 8)$, the initial state is $(k_1, k) = (144, 84)$ and for a targeted $DEO_{\Delta} = (4, 7)$, the initial state is $(k_1, k) = (15, 76)$.



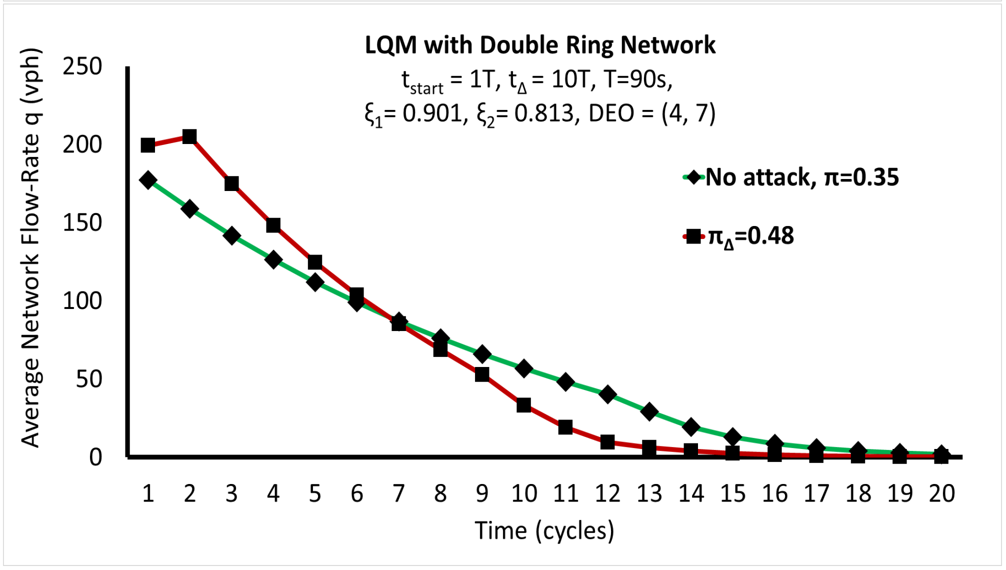
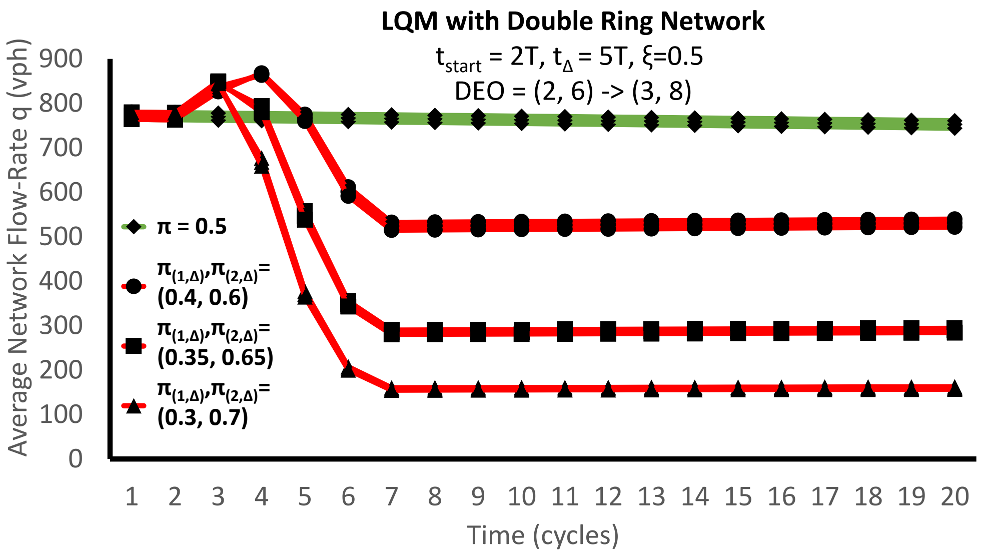
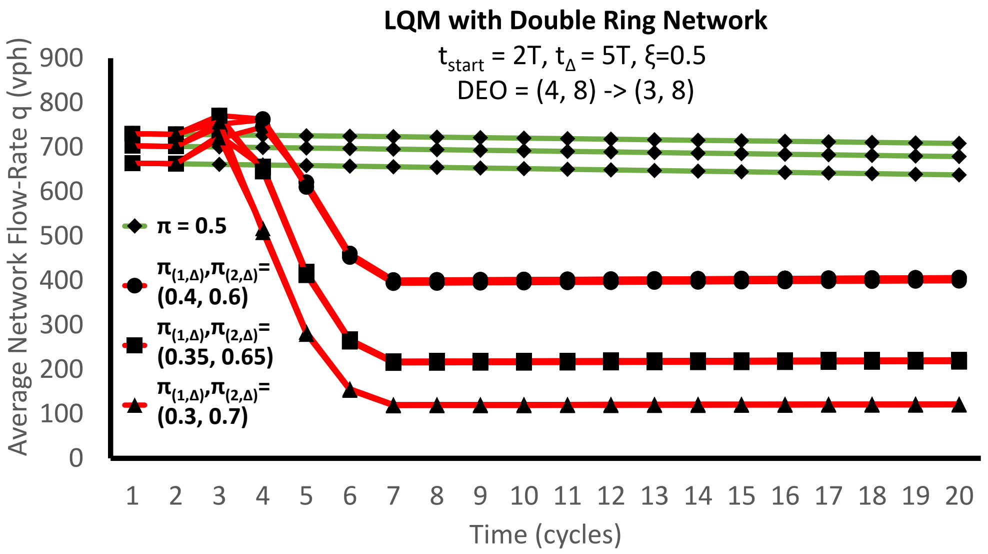


Figure 3.13: Non-State-Changing Attack Model Simulations with Randomly Selected and Asymmetric Retaining Ratios ξ_1= 0.901 and ξ_2= 0.813. For a targeted DEO_Δ = (3, 8), the initial state is (k_1, k) = (144, 84) and for a targeted DEO_Δ = (4, 7), the initial state is (k_1, k) = (15, 76).



(a) Attacks on three initial states with DEO = (2, 6).



(b) Attacks on three initial states with DEO = (4, 8).

Figure 3.14: Simulations with LQM and Double Ring Road Network of state-changing attacks. Attack timing settings are $t_{start} = 2T$ and $t_\Delta = 5T$.

where $t_{start} = 2T$, $t_\Delta = 5T$, and $t_{start+\Delta} = 7T$. In Figure 3.14b, the initial states have a DEO of (4, 8) with same initial signal settings and simulation settings as above.

Notice that the overall impacts are greater on initial states with DEOs of (4, 8) rather than attacks on those with (2, 6). This is because of the order of phases and that $\pi_{1,\Delta} < \pi_{2,\Delta}$. If $\pi_{1,\Delta} > \pi_{2,\Delta}$ instead, the effectiveness of increasing the duration and modifications would be reversed. To give a better idea on the impacts of these state-changing attacks, Table 3.3 is referred. The rows correspond to the different combinations of initial states, DEOs, and attack timing settings while the columns refer to different modifications. From just a few cycles of modified green time ratios, an attack can vary from 37% to a 99% drop in average flow.

	Attack Modifications		
Initial State and Timing Parameters	**$\pi_{(1,\Delta)}, \pi_{(2,\Delta)}$= 0.4, 0.6**	**$\pi_{(1,\Delta)}, \pi_{(2,\Delta)}$= 0.35, 0.65**	**$\pi_{(1,\Delta)}, \pi_{(2,\Delta)}$= 0.3, 0.7**
DEO_o = (2, 6) t_{start} , t_Δ = 2T, 5T	-334.45 42%	-543.34 68%	-655.31 83%
DEO_o = (2, 6) t_{start} , t_Δ = 1T, 11T	-691.15 37%	-767.35 66%	-786.59 81%
DEO_o = (4, 8) t_{start} , t_Δ = 2T, 5T	-256.29 45%	-399.25 70%	-475.87 83%
DEO_o = (4, 8) t_{start} , t_Δ = 1T, 11T	-500.41 86%	-552.56 97%	-565.73 99%

Table 3.3: Average impact metric $q_\Delta - q_0$ (in vph) and average drop in flow (in percentage) for the considered State-Changing Attack models ($\xi = 0.5$, $\pi = 0.5$, $DEO_\Delta = (3, 8)$).

3.5.2 LQM with Grid Network

The same attacks in Section 3.2.3 are evaluated using LQM and a grid network (like that in Figure 3.4) where each ring number corresponds to each direction (E-W and N-S) at each intersection and cars leaving the network are added back to their respective entrances to

DEO of Initial State (π=0.5 & T=90s)	Number of Tested Initial States and Range of (k1, k) Values
DEO$_o$ = (2, 6) ***ξ=0.5***	100 (72, 75) – (90, 84)
DEO$_o$ = (4, 8) ***ξ=0.5***	310 (91, 85) – (150, 150)
DEO$_o$ = (3, 8) ***ξ=0.75***	1520 (2, 76) – (148, 149)
DEO$_o$ = (4, 7) ***ξ=0.75***	1471 (118, 76) – (150, 149)

Table 3.4: Experimental Setup: Number of different initial states according to range of density values (k_1, k) values per each type of pair of DEO and turning ratio combination studied in the attack models (both Non-State Changing and State-Changing).

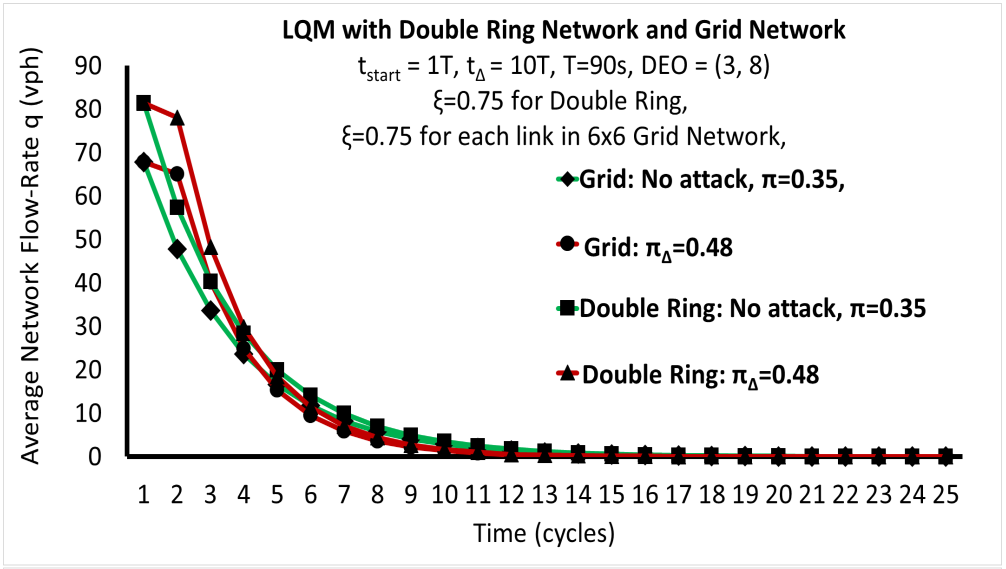
maintain the overall density (more details are in [134, 89]).

The grid network simulation results for the 4x4 grid network (with 32 links) are not provided since the results from the Double Ring Road Network and the grid network simulations on average are similar to each other with above 95% accuracy. The similarity was computed via the L1 norm and by using the grid network simulation results as the nominal values.

The LQM is also simulated with a larger 6x6 grid network with 72 links and simulation results are shown in Figure 3.15 for an initial density (k_1, k) = (144, 84) (same initial state as results in Figures 3.12 and 3.13). The top graph refers to simulations when the link retaining ratios are randomly selected (e.g., $\xi_i \in [0.51, 0.99]$) at the beginning but remain the same throughout the simulation, and the bottom graph refers to simulations when they are equal to 0.75 (e.g., ξ=0.75) using $\pi = 0.35$ and $\pi_\Delta = 0.48$). For this case study, it is clear that from the top graph, the average grid network flow is slightly less than the average Double Ring Road Network flow. From the bottom graph, one may observe that the overall grid network average flow is higher (almost double) for random retaining ratios with respect to the Double Ring Road simulation data. It is clear that the randomness of retaining ratios will have a significant effect on the understanding of traffic behaviors. Importantly, however, is that no matter the retaining ratio settings, the expected asymptotic behavior of the grid network simulations match that of the Double Ring Road Network, matching the analysis of possibly fixed asymptotic gridlock states.

Lastly, the attack impact on the convergence time is similar despite the different network models and retaining ratios. With respect to the Double Ring Road Network when $\xi = 0.75$, the attack impact is $t_{conv} - t_{conv,\Delta}$ =16T-12T=4T and 16T/12T = 1.3x convergence speedup. With respect to the grid network when $\xi = 0.75$ for each link, the attack impact is $t_{conv} - t_{conv,\Delta}$ = 15T-12T=3T and 15T/12T = 1.25x speed up, which is quite close to 1.3x! And finally, when $\xi \in [0.51, 0.99]$ for each link in the grid network, the attack impact is $t_{conv} - t_{conv,\Delta}$ = 22T-18T=4T and 22T/18T = 1.2x speedup! Thus, although the exact times of the convergence is different for the grid network with random retaining ratios per each link, the actual impact on the convergence is similar (10% average distance between each other and same overall behaviors) for all three cases, showing the value of the attack models and their analysis.

It is true, however, that the dependencies and connections are definitely not negligible and some subtle behaviors in a grid network with microscopic modeling behaviors and/or other more dynamic road networks may be missed. The Double Ring Road Network Model abstracts the grid network well since both models follow the same inflow and outflow definitions for the intersections of roads and because the grid network is assumed to be a closed network with periodic boundary conditions. A noticeable difference is that certain unstable traffic behavior patterns may not be observable in the grid network simulations. This means that Poincare Maps are not always usable to clearly define the unstable state traffic behaviors in experiments or practice. Nevertheless, in experiments or in practice, one may use the defined DEOs to identify if a state is potentially in an unstable state or not, providing the



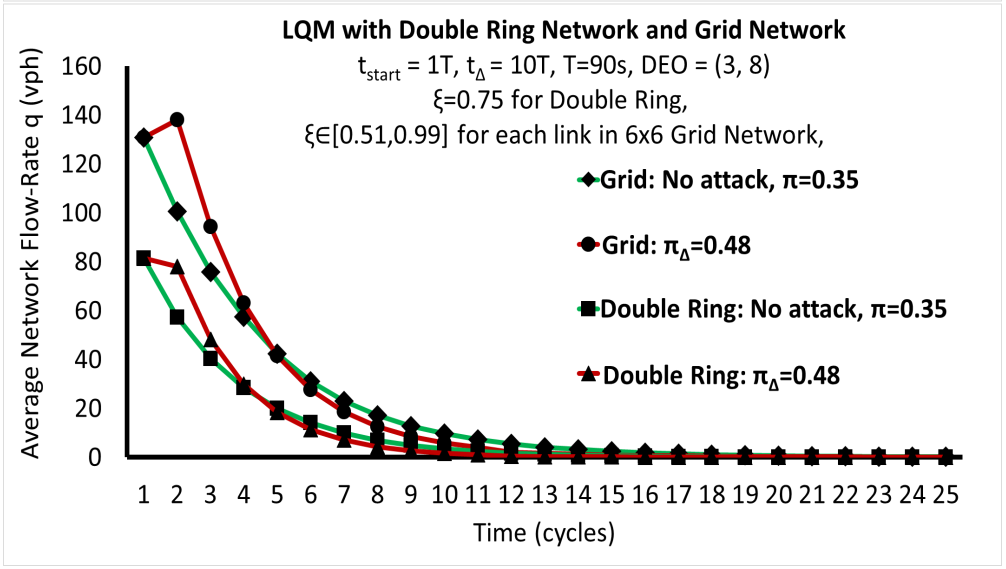


Figure 3.15: Non-State-Changing Attack Simulation Result Comparisons Between Double Ring Road Network and 6x6 Grid Network. Top: Grid Network with All Link Retaining Ratios $\xi = 0.75$. Bottom: Grid Network with Randomized Link Retaining Ratios $\xi \in [0.51, 0.99]$.

possibility to detect or analyze the impact of a potential attack.

In the previous work [89, 88], it has been shown that the results of simulating LQM with the double ring road and those with the grid network are still quite similar for when symmetric retaining ratios are assumed, signal settings and initial densities. Yet when these are randomized throughout the grid network, only some slight differences have been discovered in the analytical results (e.g., with lower average densities, the grid network will converge to gridlock with random retaining ratios).

Compared to other system and attack models, the models offer unique insights with respect to overall average network behaviors. It is built from simpler differential equations and even from Poincare Maps, which are extremely challenging to derive in traffic modeling theory. Through them, novel traffic network behaviors and insights within the context of a methodology for attack modeling and analysis on connected fixed time traffic control systems have been discovered. Through this work, more rigorous ITS security models, analysis and design methods, and simulation tools may be developed to ensure the security and safety of promising ITS use cases.

3.6 Summary

This chapter presented a methodology to model attacks on connected fixed-timing traffic control systems and evaluate their impacts on traffic networks such as grid networks in Manhattan. The methodology is built upon the Link Queue Model (LQM) which is an abstracted form of the Cell Transmission Model (CTM). However, it remains as accurate and more efficient to simulate than the CTM. For certain settings, Poincare Maps can be analytically and numerically derived to quickly estimate the performance of the system given the initial state and settings. Otherwise, the LQM and Double Ring Road Network are used to simulate either non state-changing or state-changing attacks on intersections or grid networks and evaluate their impacts. A simple attack that modified the green time ratios from $(0.5, 0.5)$ to $(0.35, 0.65)$ for 5 cycles could potentially cause an average drop in flow rate of 66%. Under certain settings, the attack could even cause a 99% drop in average flow (gridlock)!

Lastly, the proposed attack modeling and impact analysis methodology can easily be built upon with more complex attacks (e.g., combinations of small changes, turning ratio modifications) and metrics (e.g., deviation from optimal state). In turn, these attacks may be used as part of a security analysis tool to come up with a robust and resilient traffic control system design. In such a traffic control system, attack detection and mitigation may be possible through additional computing devices and sensors integrated with control logic that can predict or identify a potential attack at any given moment.

Chapter 4

Attack Modeling Methodology and Taxonomy for Intelligent Transportation Systems

4.1 Introduction and Related Work

Massive deployment of embedded systems including various sensors, on-board and road-side computing units, wireless communication among vehicles and infrastructure, and intelligent algorithms are changing the transportation sector [74, 150, 77, 23, 200]. Due to these technologies, engineers are able to provide autonomy and connectivity in transportation control systems. Therefore, this new paradigm, known as Intelligent Transportation Systems (ITS), is bringing new opportunities to solve transportation system challenges regarding traffic congestion, energy waste, vehicle emissions, and traffic accidents [267]. Today, advanced sensors and wireless vehicular communication (V2X) enable advanced algorithms for traffic management such as autonomous control, Connected Adaptive Cruise Control (CACC), collision detection and avoidance, Advisory Speed Limit (ASL), route guidance, and more [118].

As discussed in works [48, 113, 148, 11, 117], with these newer technologies come unforeseen safety and security concerns. Some of these concerns were recently revealed when traffic controllers used in almost all of the states in the US were found to be remotely hackable and controllable by an attacker [46, 48]. In addition to traffic control systems, connected autonomous vehicles provide many security and safety concerns, where effects of attacks on the peripherals or the Electronic Controller Units (ECUs) may cause congestion, but more importantly may endanger passengers and passersby [264, 79, 210, 263].

4.2 Overview and Contributions

This chapter presents a novel attack modeling methodology and taxonomy regarding potentially targeted components in an ITS. In addition, unique attack impact metrics and evaluations with different car-following models and simulation tools are provided use case of a V2X Advisory Speed Limit control application - which has never before been studied in terms of security, nor thoroughly implemented/studied in general. Implementations of an ITS application simulation may vary substantially and therefore establishing a foundation to do so is timely and critical in this rapidly developing technological age. Nevertheless, although the context of the chapter is with respect to ASL, the methods and approaches may be applied to other ITS use cases with other parameters as well.

The following items will be the order of contents of this chapter and also the summary of the contributions:

- Design and implementation of an ITS use case known as V2X Advisory Speed Limit control.

- Usage of two different simulation architectures composed of the Ring Road Network Model with 1) Newell's Car-Following Model with Bounded Acceleration (BA-Newell Model) and Matlab and 2) Intelligent Driver Model and Veins.
- Meaningful performance and attack impact metrics that may be transferable in different use cases, car-following models, and simulation tools.
- Creation of attacker and attack modeling taxonomy to serve as a guide for ITS security analysis.
- Evaluation of attack impact metrics using various attacker profiles, attack types, and timing parameters.
- Providing open source code (*https://github.com/AICPS/ITSAttackModeling/*) for both of the architectures to allow others in the ITS community to utilize or improve upon the work.

4.3 ITS Attack Modeling Literature Review

With respect to attack modeling in traffic control systems and connected vehicles, past studies have studied attacks on unique use cases and network models [55, 95, 164]. In contrast, the Single Ring Road Network Model is used for its simplicity and ability to abstract more complex network and system models. The attack vectors proposed in these works are the same as those in the past studies, aka, the methods to exploit vulnerabilities for an attacker to gain access in order to inject their attack. These vectors may target various components and subsystems of the ever-growing ITS. Further, an attack vector on one such component/subsystem will tend to lead to effects on other components/subsystems due to the connectivity between them. More interesting to us in this work, is how those attack vectors may be used to modify the control variables of an ITS application to achieve various objectives depending on attacker profiles, timing, attacker budgets, and attack costs.

Historically, ASL has been performed using only physical signs on the road to help reduce speeds. With the advent and adoption of V2X and 5G, traffic management applications like ASL will be strongly improved. In the paradigm of ITS, ASL is one of the key upcoming applications whose impacts have not yet been analytically or experimentally studied in terms of security [200]. ASL focuses on taking information from induction loop sensors and the connected vehicles arriving to an intersection from upstream to then advise the connected vehicles the maximum velocity (v_{asl}) that they should follow. The maximum velocity, v_{asl}, is computed so that each vehicle will always (ideally) arrive to the intersection at the green time. When a majority of vehicles follow such velocities for a certain stretch of road, the average amount of waiting time in a trip (e.g. from one end of the road to past the intersection) is drastically reduced.

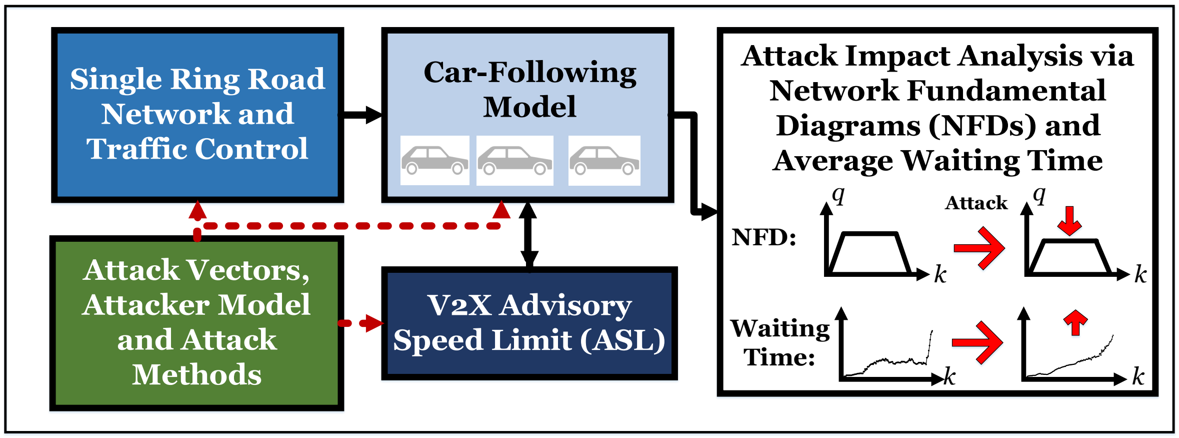

Single Ring Road Network and Traffic Control
Car-Following Model
Attack Impact Analysis via Network Fundamental Diagrams (NFDs) and Average Waiting Time
Attack Vectors, Attacker Model and Attack Methods
V2X Advisory Speed Limit (ASL)
NFD:
q
k
Attack
Waiting Time:

Figure 4.1: Methodology overview. The dashed red arrows correspond to the points of possible attack injections and the solid black arrows are the dependencies between the models. The Car-Following Model and ASL blocks may be switched with equivalent components to evaluate other models and ITS applications. Here, q is the average network flow in veh/s and k is the average vehicle density in veh/m).

4.4 Threat Modeling of an ITS

4.4.1 Attack Vectors

Many embedded systems within the subsystems of ITS typically have several vulnerabilities that may be exploited remotely once wireless communication is introduced to them, e.g., Dedicated Short-Range Communications (DSRC/IEEE 802.11p), Wireless Access in Vehicular Communications (WAVE/IEEE 1609), cellular (4G, 5G), Bluetooth [241, 222, 154, 95, 96, 47]. Besides this, attackers can directly alter the software in internal vehicular hardware, such as Electronic Controller Units (ECUs), via the On-Board Diagnostic (OBD) port and infotainment system [52, 53]. Additionally, peripherals such as sensors including induction loop detectors, tire speed (Hall effect) sensors, GPS sensor/GNSS receivers may all be targeted for manipulation or spoofing [210, 47, 125, 239, 235].

These all may be exploited to perform an attack that will impact the average network flow and waiting time. The attacker may vary from a teenager performing hacks for fun [170], to an angry employee [286], to terrorist organizations. Attacker objectives may vary from slowing down traffic for a single vehicle to macroscopic scale traffic congestion. Although the work in this chapter is at a macroscopic scale, if desired, the attack models and impact metrics may be configured to focus on an individual vehicle or a smaller subgroup of vehicles.

4.4.2 Attaining System State Knowledge

With or without access to the wireless network, the attacker may attain knowledge about the current state of the system,such as the current timing plan configuration, the physical state of the intersection, or vehicle speeds and positions. Without network access, the attacker may easily use off-the-shelf equipment (e.g., RADAR, Infrared, wireless magetometers, acoustic sensors) or their observations because the traffic intersection is in a public space. With access, they may use the measurements from existing vehicle and system sensors. It is also assumed that, with the introduction of more technology (i.e., image processing, smarter sensors), it will be even easier to accurately estimate the current system state. Works [96, 47, 46] describe in more detail how these are possible in a realistic setting.

4.5 System Modeling

As mentioned, two alternate architectures are created to implement the attacker and attack models with and to evaluate the usefulness of the impact metrics.

4.5.1 Traffic Network Model

In both architectural implementations, the Single Ring Road Network Model is used because of its effectiveness in deriving a realistic metric representing the average performance of the traffic network model (whether just a single junction or an entire grid network). In [251], researchers experimented and studied traffic behavior using a single ring road. From their studies, they observed that after certain numbers of vehicles (i.e., called critical densities), the asymptotic traffic behavior changes despite no existing bottleneck in the road. Studies have shown that observed behaviors regarding this model match with empirical studies on actual traffic as well [251, 141, 68].

Figure 3.3 presents a real-life experimental setup of the model using robotic cars equipped with sensors and Arduino boards [141]. The logic used by these robots based on the sensors is called the Optimal Velocity Model (OVM) and is similar to car-following logic used in this chapter. Despite the inaccuracies of the sensors and imperfect execution of the code, the experimental results from this work positively emulated the empirical and theoretical results from previous studies and may be used for experimentation of car-following models and connected vehicle-based control algorithms.

4.6 V2X Advisory Speed Limit (ASL)

In general, the speed limit and maximum speed is defined as the free flow speed, $v_f = 15$ m/s. However with V2X ASL, the traffic control system takes information from induction loop sensors and connected vehicles to compute advisory speed limits (v_{asl}) for each connected vehicle instead. This will ideally permit vehicles to arrive at the intersection when the light is green or yellow, rather than at a red light. Hence, ASL substantially reduces the average waiting time aka stopping time, improves the overall average network flow-rate, and subsequently lowers environmental costs from greenhouse gas emissions and even improves driver attitudes [267].

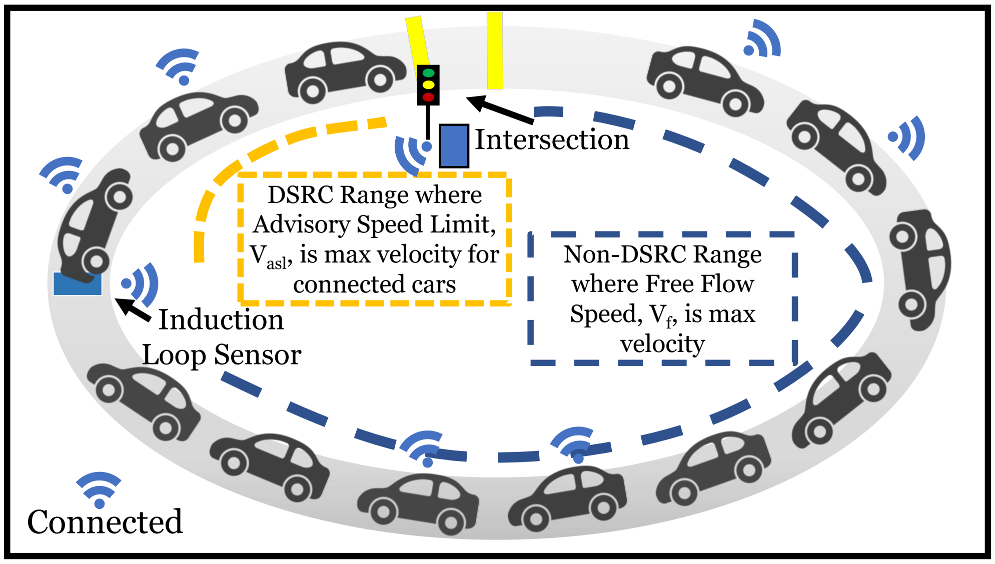

Intersection
DSRC Range where Advisory Speed Limit, V_{asl}, is max velocity for connected cars
Non-DSRC Range where Free Flow Speed, V_f, is max velocity
Induction Loop Sensor
Connected

Figure 4.2: *Single Ring Road Network Model* and ASL control system. Vehicles will receive ASL velocities via DSRC to avoid arriving at the intersection during the red time and to reduce overall waiting time.

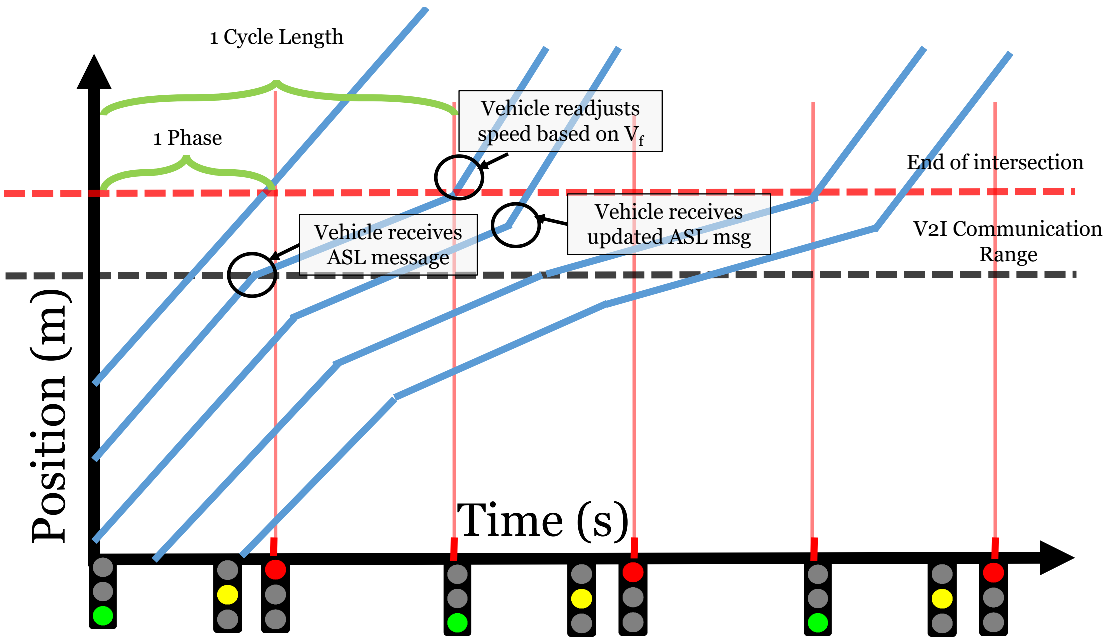

1 Cycle Length
1 Phase
Vehicle readjusts speed based on V_f
Vehicle receives ASL message
Vehicle receives updated ASL msg
End of intersection
V2I Communication Range
Position (m)
Time (s)

Figure 4.3: Representation of vehicles' path traces (diagonal solid blue lines) when ASL is implemented at a junction (end of intersection is start of ring road in the models). If ASL was not implemented, vehicles would arrive at the red times and cause other vehicles to wait until the light turns green.

4.6.1 Vehicles That Have Not Received Advisory Speed Limit

For each connected vehicle n that comes into the communication range, the traffic controller will receive the vehicle's velocity and position information and compute and transmit a unique ASL, $v_{asl}(t,n)$ to it. This $v_{asl}(t,n)$ replaces v_f throughout the remainder of the trip through the road (until it reaches the intersection) of a connected vehicle that received it. Vehicles without wireless communication will update their positions and velocities using the car-following position and velocity update equations with v_f as the maximum speed. Unlike the connected vehicles using ASL, the non-connected vehicles will typically end up hitting a red light and therefore cause other vehicles behind them to wait until the next green light. In simulations, how many vehicles will be connected or not may be chosen based on a Market Penetration Rate ($MPR \in [0,1]$) variable, but for now $MPR = 1$ is used.

Because $\Delta n = 1$ is used, $\Delta t = \tau\Delta n = 1.5 >> \bar{t}_{comm}$, and therefore it is assumed that all packets will have a minimum static one-way trip delay of $t_{comm} = \Delta t$. When vehicles arrive at $L - L_{DSRC}$, that is the first time they are able to begin the ASL procedure. The traffic control system receives the vehicles' Basic Safety Messages (BSMs) [150, 77, 23] and replies to them with unique ASL messages containing v_{asl} (as represented by Figure 4.2). When the traffic controller receives a BSM from a vehicle n, it will compute that vehicle's estimated time of arrival to the intersection, $ti(t,n)$, based on the number of detected vehicles that are in front, $vif(t,n)$ (using a loop sensor positioned at $L - L_{DSRC}$ m on the road). Then, with the kinematics data from the nth vehicle's BSM and $vif(t,n)$, the controller will compute $ti(t,n)$ using the following equation:

$$ti(t,n) = min(t + \frac{L_1 - \tilde{x}(t,n)}{v_f},\ t + \frac{vif(t,n)}{sr}) \tag{4.1}$$

where $sr = 1800/3600$ vehicles per second is the typical intersection service rate.

Afterward, v_{asl} may be computed using the expected arrival time (ti) of a vehicle using Equation 4.2.

$$\begin{aligned} v_{asl}(t,n) = & \\ & min(v_f, \frac{L_1 - \tilde{x}(t,n)}{ti(t,n) - t}) \\ & where\ \tilde{x}(t,n) = x(t - t_{comm}, n) + x_{err} + \\ & v(t - t_{comm}, n)\tilde{t}_{comm} \end{aligned} \tag{4.2}$$

where parameter and variable names and definitions are provided in Table 4.1.

4.6.2 Vehicles That Have Received an Advisory Speed Limit

Vehicles that have received their first ASL velocity may/may not need to update their ASL velocity based on the current state of the roads and traffic junction.

Cycle Length (T)	60s
Green Time (G)	24 s
Yellow and All-Red Time (Y)	6 s
Red Time (R)	30s
Effective Green Time Ratio (π)	0.5
Simulation Time (t_{sim})	1200 s
Free Flow Velocity (V_f)	15 m/s
Road Length (L)	900 m
Intersection Length (L_{int})	10 m
Length of Road Before Intersection (L_1)	890 m
DSRC Range (L_{DSRC})	300 m
Induction Loop Sensor Position ($L - L_{DSRC}$)	600 m
Jam Distance (ρ)	7 m
Vehicle Length	5 m
Minimum Headway Gap	2 m
Time Gap (τ)	1.5 s
Vehicle Step (Δn)	1
Time Step (Δt)	$\tau \Delta n = 1.5s$
Critical Density 1 ($k_{c,1}$)	15 veh/m
Critical Density 2 ($k_{c,2}$)	76/L veh/m
Capacity Critical Density (k_c)	31/L veh/m
Capacity Flow of BA-Newell Model ($C = k_c vf$)	.508 veh/s
Capacity Flow of Intelligent Driver Model (C_{IDM})	.4 veh/s
Jam Density (k_j)	128/L veh/m
Shock Wave Velocity (w)	4.67 m/s
Average Network Flow (q)	0-C
Waiting Time Ratio (r_{wait})	[0,1]
Number of Vehicles (N)	0-$k_j L$
Vehicles in Front (vif)	0-vif_{max}
Maximum Number of Vehicles in Front (vif_{max})	L_{DSRC}/ρ
Estimated Intersection Arrival Time (ti)	[t, t_{sim}]
GPS Error x_{err}	[-5, 5] m
DSRC Delay Mean t_{comm}	max($\Delta t, 0.1s$)
Market Penetration Rate (MPR)	1.0

Table 4.1: System Model Variables and Parameters

If the expected arrival time $ti(t, n)$ is greater than current time step t, the traffic controller will send the vehicle an updated v_{asl} according to the vehicle's positional information in the latest BSM (see Equation 4.3).

$$v_{n,asl} = min(v_f, \frac{L_1 - x(t, n)}{ti(t, n) - t}) \tag{4.3}$$

When a vehicle's expected arrival time $ti(t, n)$ is less than or equal to t, its arrival time $ti(t, n)$ must be updated. First, the maximum velocity is reset back to v_f, then its new intersection arrival time is re-estimated with it. The new arrival time is computed as follows:

$$ti(t, n) = t + \frac{L_1 - x(t, n)}{v_f} \tag{4.4}$$

Any time that the newly computed arrival time $ti(t, n)$ might be within the red signal phase, the traffic control system detects it, recomputes it so that it is equal to the start of the next green time, and sends out a new v_{asl} based on it. The formula to do so is shown in Equation 4.5.

$$ti(t, n) = ti(t, n) + T - mod(ti(t, n), T) \tag{4.5}$$

where $mod(a, b)$ is the modulus function equivalent to *a modulus b*.

Note that there are various ways that V2X ASL may be implemented and this is one of them. In fact, before the work in this chapter, there are no other works that have utilized this kind of ASL, which attempts to maximize the usage of V2X to reduce overall waiting time and improve traffic flow.

4.6.3 Architecture 1: Matlab and Newell's Car-Following Model with Bounded Acceleration

In Architecture 1, the entire implementation is written in Matlab [2] and therefore full control over the car-following model, the network model, the signal settings, and vehicle behaviors is provided. However, the implementation lacks realistic wireless communication networking (links, packets, etc.) and physical channel modeling that Architecture 2 has. Additionally, it does not consist of detailed sensor definitions (e.g., induction loops) or visualizations.

In Matlab, Newell's Car-Following Model with Bounded Acceleration (the BA-Newell model) is implemented, which is relatively simple, has thorough analytical properties, and safety guarantees (unlike more commonly used car-following models) [139, 138]. In this model, vehicles abide by a bounded acceleration but there is no bounded deceleration. Signal settings and model settings are provided in Table 4.1. A 100% aggressive vehicle population is considered, where each vehicle decides to go through the intersection if they are able to do so at their current velocity [138].

4.6.4 Architecture 2: Veins and Intelligent Driver Car-Following Model

For the second architecture, a renowned ITS simulation tool called Veins [248, 249] is used. Veins is an open-source ITS simulation tool that integrates its own V2X network stack implemented within the communication network simulation tool OMNeT++ [275, 276] with the traffic network, car-following, and vehicle simulation tool SUMO [31]. Via SUMO's

Traffic Controller Interface (TraCI), Veins (programmed in OMNeT++) may access various parameters and variables of SUMO. Since the wireless communication considers aspects such as delay and fading in realistic environments and because SUMO is a dedicated and well-developed tool for traffic simulation, Veins is a valuable tool for the purposes to test an ASL use case and the novel attack models. However, in addition to subtle elements that both SUMO and OMNeT++ have (such as randomness in car following and messaging, collision detection and teleportation), it suffers from its incredible complexity and daunting documentation for those outside of the traffic modeling study field.

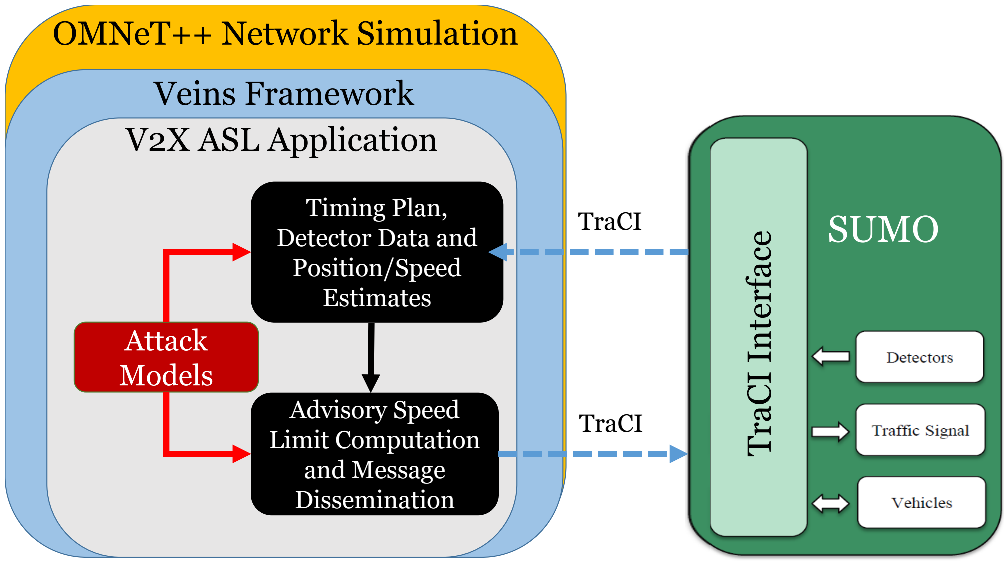

OMNeT++ Network Simulation
Veins Framework
V2X ASL Application
Timing Plan, Detector Data and Position/Speed Estimates
Attack Models
Advisory Speed Limit Computation and Message Dissemination
TraCI
TraCI
TraCI Interface
SUMO
Detectors
Traffic Signal
Vehicles

Figure 4.4: Architecture 2 comprising of SUMO (car-following and traffic network simulator) and OMNeT++ (communication simulator) connected with Veins Framework through the TraCI (Traffic Controller Interface) API. The ITS use case, V2X Advisory Speed Limit, and the Attacker/Attack Models are implemented within Veins.

A traffic network is designed and developed that emulates both a realistic single junction arterial network and the behavior of the Single Ring Road Network. Since Veins does not include the BA-Newell Model, the popular Intelligent Driver Model (IDM) with $\delta = 32$ is used instead (to make it closer to the BA-Newell Model's behaviors) [266]. The IDM capacity ($C_{IDM} \approx 0.4$) differs from that of the Triangular Fundamental Diagram ($C \approx 0.5$), which the BA-Newell Model follows [266], and thus the q values must be normalized with C_{IDM} and use a 2.5 s / veh saturation headway (aka Service Rate) for the ASL update methods.

4.7 Attacker and Attack Modeling Methodology

4.7.1 Targeted Components and Control Variables

Considering the ITS components used in V2X ASL helps narrow down the possible data-based variables that the attackers may target to reduce and reverse ASL effects. Sensor-based attacks are of primary focus, but the variables also be attacked via a tampered OBU or spoofed BSMs under certain circumstances.

Vehicle to Infrastructure (V2I) Communication Channel

The variable of interest for this medium is the transmission delay itself, t_{comm}, leading to the ITS in using an incorrect position or velocity.

Induction Loop Sensor

The expected arrival time, $ti(t, n)$, will be adversely affected an incorrectly detected number of vehicles in front of a vehicle, vif, due to physical tampering or packet spoofing.

GPS sensor / GNSS receiver

An exploit on a vulnerable GPS sensor/GNSS receiver that targets the variable of interest $\tilde{x}$.

Speed / Hall Effect Sensor

Hall effect sensors are located on vehicle tires that are used to measure speeds. Hence, the velocity value, v, that is computed by the vehicle and transmitted to the traffic control system may be targeted and perturbed.

Traffic Controller

Last, but not least, the v_{asl} itself may be modified through exploitation of a traffic controller itself. Since the overall effects from changing v_{asl} through indirect attacks on other mentioned components may be observed and it is more obvious to observe a direct change to v_{asl}, it is left out of this work.

4.7.2 Performance Metrics

Average Network Flow Impact

A Network Fundamental Diagram (NFD) maps an asymptotic average network flow q (product of average network density k and average network velocity $\bar{v}$) with a current traffic state (number of vehicles N or average density k) and is useful in practice for traffic engineers and traffic control systems because they may be quickly used to estimate the current traffic control system performance.

To compute an NFD, the combined car-following and network model with the ITS use case must be simulated for several cycles to compute q corresponding to each defined N, with step $\Delta N = 10\ veh$. A simulation of 1200 seconds is quick, yet sufficient enough, to compute $q = \bar{v}(N/L)$, where $\bar{v}$ is the space-mean velocity for all N vehicles on road of L length. Of notable importance, is how q compares to the capacity, C. And thus, from hereon after,

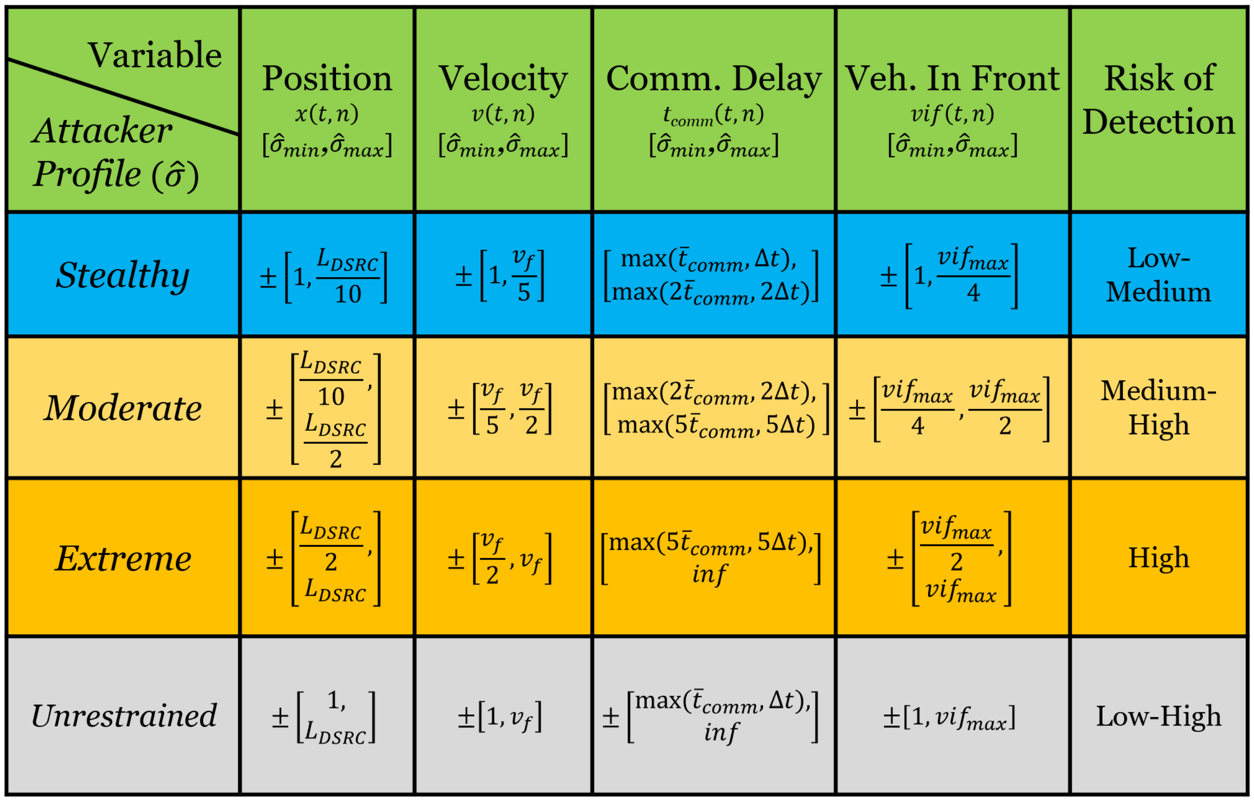
q/C will be used to evaluate the traffic flows and have a better understanding of the system performance.

To compute the attack impact on the average network flow involves taking the difference between the $\hat{q}/C$ after the attack and the original q/C for the same corresponding number of vehicles N (i.e., $\hat{q}_{\Delta} = \frac{q-\hat{q}}{C}$). Another useful property is the trend of the attack impact, and thus the Rate of Change (RoC) is computed between two consecutive vehicle counts of the average network flow impact as $\hat{RoC}_{q\Delta} = \frac{\hat{q}(N_2)_{\Delta} - \hat{q}(N_1)_{\Delta}}{N_2 - N_1}$

Average Waiting Time Ratio Impact

A relative Average Waiting Time Ratio r_{wait} is also calculated. It is the ratio of average time steps that $v < 0.1 m/s$ out of the average trip duration time steps of all vehicles in the network (N), rather than the more frequently used absolute Waiting Time (total time that $v < 0.1 m/s$). More uniquely, r_{wait} consists of normalization using the Average Waiting Time Ratio when there is no ASL, denoted as r_{wait,No_ASL}, of the corresponding architecture. This is so that a fairer comparison between the two architectures may be made[1]. The Average Waiting Time Ratio is $r_{wait} = (t_{wait}/\bar{t}_{wait})/r_{wait,No\_ASL}$ and the impact on the Average Waiting Time Ratio from an attack is $\hat{r}_{wait\Delta} = \hat{r}_{wait} - r_{wait}$.

The Waiting Time Attack Impact is computed by taking the difference of the metric before and after attack for the same corresponding number of vehicles N (e.g., $\hat{r}_{wait\Delta} = \hat{r}_{wait} - r_{wait}$). The RoC of the normalized waiting time impact also serves as a metric: $\hat{RoC}_{wait\Delta} = \frac{\hat{r}_{wait\Delta}(N_2) - \hat{r}_{wait\Delta}(N_1)}{N_2 - N_1}$.



Variable / *Attacker Profile* ($\hat{\sigma}$)	Position $x(t,n)$ $[\hat{\sigma}_{min}, \hat{\sigma}_{max}]$	Velocity $v(t,n)$ $[\hat{\sigma}_{min}, \hat{\sigma}_{max}]$	Comm. Delay $t_{comm}(t,n)$ $[\hat{\sigma}_{min}, \hat{\sigma}_{max}]$	Veh. In Front $vif(t,n)$ $[\hat{\sigma}_{min}, \hat{\sigma}_{max}]$	Risk of Detection
Stealthy	$\pm\left[1, \frac{L_{DSRC}}{10}\right]$	$\pm\left[1, \frac{v_f}{5}\right]$	$\left[\begin{matrix}\max(\bar{t}_{comm}, \Delta t), \\ \max(2\bar{t}_{comm}, 2\Delta t)\end{matrix}\right]$	$\pm\left[1, \frac{vif_{max}}{4}\right]$	Low-Medium
Moderate	$\pm\left[\begin{matrix}\frac{L_{DSRC}}{10}, \\ \frac{L_{DSRC}}{2}\end{matrix}\right]$	$\pm\left[\frac{v_f}{5}, \frac{v_f}{2}\right]$	$\left[\begin{matrix}\max(2\bar{t}_{comm}, 2\Delta t), \\ \max(5\bar{t}_{comm}, 5\Delta t)\end{matrix}\right]$	$\pm\left[\frac{vif_{max}}{4}, \frac{vif_{max}}{2}\right]$	Medium-High
Extreme	$\pm\left[\begin{matrix}\frac{L_{DSRC}}{2}, \\ L_{DSRC}\end{matrix}\right]$	$\pm\left[\frac{v_f}{2}, v_f\right]$	$\left[\begin{matrix}\max(5\bar{t}_{comm}, 5\Delta t), \\ inf\end{matrix}\right]$	$\pm\left[\begin{matrix}\frac{vif_{max}}{2}, \\ vif_{max}\end{matrix}\right]$	High
Unrestrained	$\pm\left[\begin{matrix}1, \\ L_{DSRC}\end{matrix}\right]$	$\pm[1, v_f]$	$\pm\left[\begin{matrix}\max(\bar{t}_{comm}, \Delta t), \\ inf\end{matrix}\right]$	$\pm[1, vif_{max}]$	Low-High

Table 4.2: Configurations for Attacker Profile, $\hat{\sigma}$. Note that the defined variable perturbation ranges ($\hat{\sigma}_{min}$, $\hat{\sigma}_{max}$) for each attacker profile are meant to serve as guides. They may be easily altered to fit a certain system design or attack analysis objective.

4.7.3 Attacker Profiles

The Attacker Profile function is denoted with $\hat{\sigma}$. Its main purpose is to define the limits to the actual perturbation value $\hat{\alpha}$. In Table 4.2,the characteristics of three different possible attacker profiles *stealthy*, *moderate*, *extreme*, are provided.

A *stealthy* attacker has the objective of keeping their risk of being detected low by subtly injecting attacks with small $\hat{\alpha}$ values that appear to be slightly extreme faults. As expected, such injections may not necessarily create a profound detrimental impact on the system. However, the longer and more frequent the attacks are, the more likely the overall application performance will asymptotically degrade on average. A *moderate* attacker injects riskier attacks, albeit still difficult to detect, to make a dent into the overall performance at a faster rate. An *extreme* attacker has no concern for risk and will (if budget permits) inject a highly noticeable perturbation to their targeted system components/variables. Finally, if desired, an attacker denoted as *unrestrained* may use the full range of the variable perturbation. The attacker profile primarily ensures that the attack perturbation values satisfy the pre-defined constraints of that profile. It is assumed that there are no constraints on the budget for the attacker. However, it can be incorporated with cost requirements as coefficients and additional functions in the attacker and attack models so that more complex attacks may be studied.

4.7.4 Attack Modeling

The attack models defined in this work are comprised of Attack Values and Attack Timing. Both Attack Values and Attack Timing may be defined as either *static* or *random*. However, Attack Timing also includes additional periodicity elements if desired.

[1]When a corresponding waiting time ratio for the no ASL case is less than %10.0, this work opts to not use it for normalization and simply take the difference for the impact. Hence values may seem much larger compared to other ones.

Attack Value

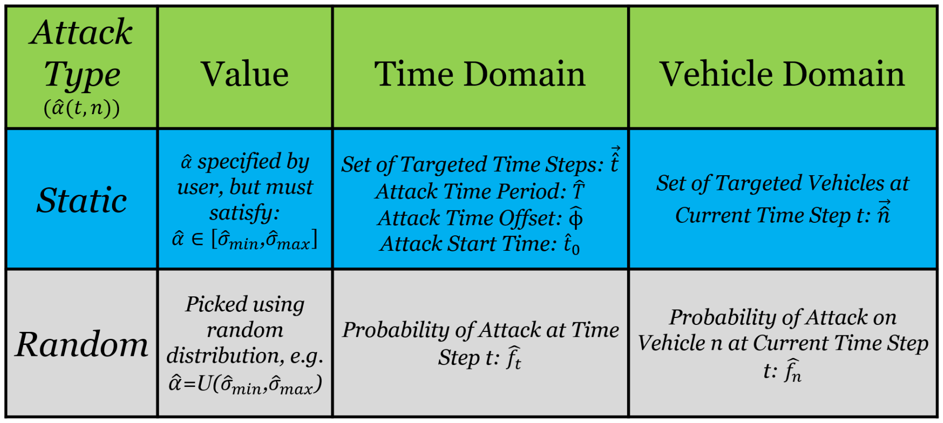
In *Static* attacks, the perturbation value remains the same. On the other hand, the *Random* attacks will continuously choose a perturbation value at random (by default, via uniform probability distribution) in the range $[\hat{\sigma}_{min},\ \hat{\sigma}_{max}]$ defined by the Attacker Profile, $\hat{\sigma}$.

Attack Timing

The attack timing is defined using the following variables: time vector $\overrightarrow{t}$, attack time vector $\overrightarrow{\hat{t}}$, attack frequency $\hat{f}_t$, vehicle vector at current time $\overrightarrow{n}$, vector of targeted vehicles $\overrightarrow{\hat{n}}$, vehicles attacked at current time $\hat{f}_n$, attack time period $\hat{T}$, time between periodic attacks $\hat{\phi}$.

For static timing, the $\hat{f}_t$ may equal $1/\hat{T}$; however, all attacks may optionally have random timing in both the time and vehicle domains. With respect to random timing, $\hat{f}_t$ and $\hat{f}_n$ are probabilities used to represent probabilistic events. A_t denotes the event that there is an attack during time t and A_n denotes that there is an attack on vehicle n. Let $P(A_t) = \hat{f}_t$, $P(A_n|A_t) = \hat{f}_n$, $P(\tilde{A}_t) = 1 - \hat{f}_t$ and $P(\tilde{A}_n|A_t) = 1 - \hat{f}_n$. As A_n is dependent on A_t, to know the full probability of an attack on vehicle n at time t (e.g. $P(A_t \cap A_n)$), it is necessary to compute $P(A_t)P(A_n|A_t)$. Therefore, $P(A_t \cap A_n) = P(A_t)P(A_n|A_t) = \hat{f}_t\hat{f}_n$ and $P(A_t \cap \tilde{A}_n)$ $= P(A_t)P(\tilde{A}_n|A_t) = \hat{f}_t(1 - \hat{f}_n)$. Note that $P(\tilde{A}_t \cap A_n) = 0.0$ and $P(\tilde{A}_t \cap \tilde{A}_n) = 0.0$.

Having these probabilistic events, one may combine them with the perturbation value to determine if there will be a perturbation or not at the current time and for the targeted vehicle. This is denoted with the expression: if $(A_t \cap A_n)$, then $\hat{\alpha}(var[i](t, n))$ remains unchanged; else, $\hat{\alpha}(var[i](t, n)) = 0$. Thus, only if the events A_t and A_n are both successful with probability $P(A_t \cap A_n) = f_t f_n$, then the overall value will be a perturbed value, else it will not be. As these events are probabilistic, one may use any kind of probabilistic



Attack Type ($\hat{\alpha}(t,n)$)	Value	Time Domain	Vehicle Domain
Static	*$\hat{\alpha}$ specified by user, but must satisfy: $\hat{\alpha} \in [\hat{\sigma}_{min}, \hat{\sigma}_{max}]$*	*Set of Targeted Time Steps: $\vec{\hat{t}}$ Attack Time Period: $\hat{T}$ Attack Time Offset: $\hat{\phi}$ Attack Start Time: $\hat{t}_0$*	*Set of Targeted Vehicles at Current Time Step t: $\vec{\hat{n}}$*
Random	*Picked using random distribution, e.g. $\hat{\alpha}=U(\hat{\sigma}_{min}, \hat{\sigma}_{max})$*	*Probability of Attack at Time Step t: $\hat{f}_t$*	*Probability of Attack on Vehicle n at Current Time Step t: $\hat{f}_n$*

Table 4.3: Configurations for various Attack Models. An Attack Model may either have static/random $\hat{\alpha}$ values and may have static/periodic/ random timing. Note: If the attack is chosen to be aperiodic, the attack time period will be the duration of the attack instead.

distribution suitable to their system/attacker/attack model design. In fact, the probability distribution may even be directly associated with a defined attacker time budget and/or attack costs.

4.7.5 Combined Attacker and Attack Model

Combining the Attacker Profile $\hat{\sigma}$ with Attack Model $\hat{\alpha}$, $\hat{\gamma}(var[i], t, n)$ is obtained (a visual representation of what the time and vehicle-based function $\hat{\gamma}(var[i], t, n)$ would look like is in Figure 4.5). After vehicle n sends a message of its kinematics, the final attacked ASL velocity v_{asl} that the RSU will send to the vehicle at time t is defined as the following:

$$\begin{aligned}\hat{v}_{asl}(t,n) =& min(v_f, \\ & \frac{L_1 - \hat{x}(t-\hat{t}_{comm}) + \hat{x}_{err} + \hat{v}(t-\hat{t}_{comm})\tilde{t}_{comm}}{(\hat{ti}(t,n)-t)})\end{aligned} \tag{4.6}$$

where for each variable var, there is a $v\hat{a}r[i](t,n) = var[i](t,n) + \hat{\gamma}(var[i], t, n)$.

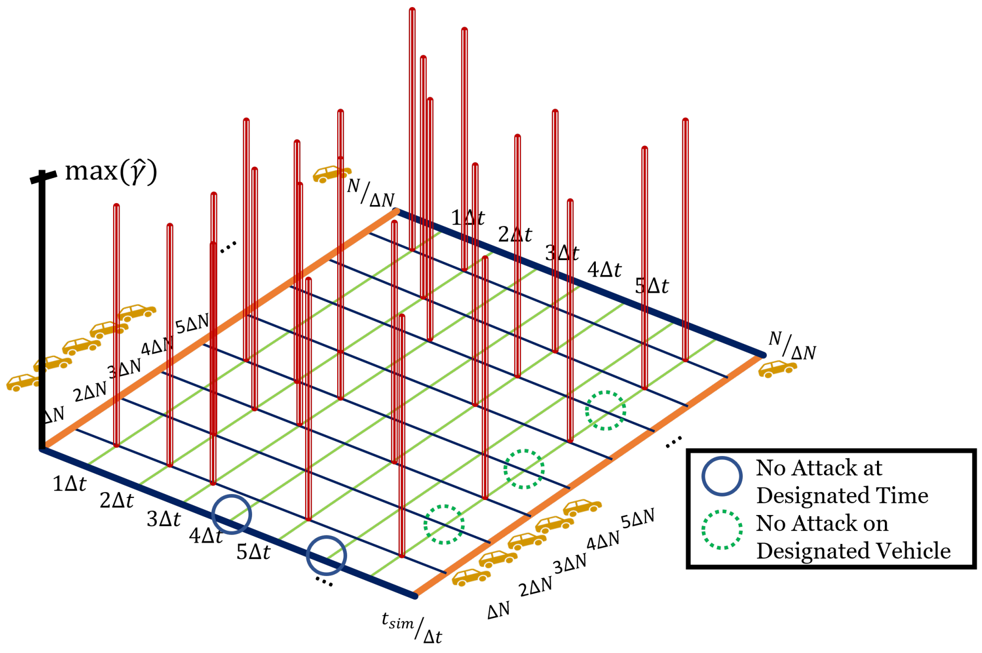


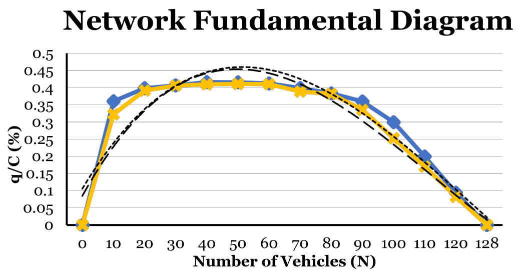
Figure 4.5: A Visual Representation of the Combined Attacker and Attack Modeling Time and Vehicle Domain-based Function, $\hat{\gamma}$ for an Extreme Attacker with Static Attack Value and Random Attack Timing.

4.8 Experimental Results

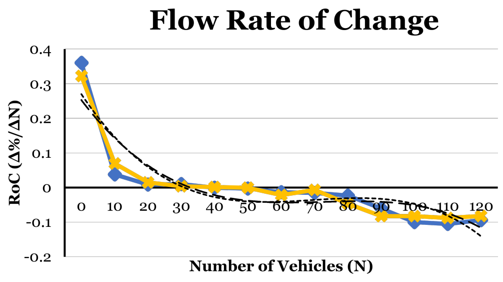
This subsection summarizes and provides analyses on some experimental results[2]. For the purposes of data visibility, low simulation time overhead (Veins), and to primarily observe and identify trends, results for each N in the following set of vehicle numbers: {0, 10, 20, ..., 110, 120, 128} are provided. Hence, ΔN is 10 veh, except for the last step (128-120=8 veh). However, the code is readily available and may be used to obtain results for any real number N within [0, 128].

Before the discussion of the attack-related results, the results when ASL is implemented and there is no attack are analyzed (see Figure 4.6). From now on, Architecture 1 is denoted with *Matlab* and a yellow line with "x"-like markers. Architecture 2 is denoted with *Veins* and

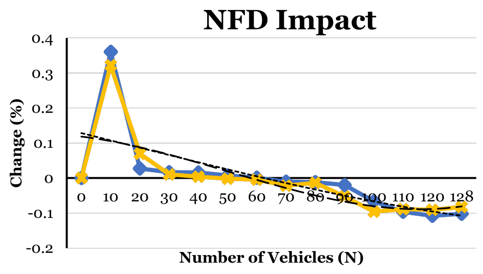
[2]Videos on several of the Veins attack modeling simulations are provided in *https://sites.google.com/uci.edu/itsattackmodelingresearch/home/v2x-advisory-speed-limit?authuser=0* and source code is on *https://github.com/AICPS/ITSAttackModeling/* to help readers better follow the chapter and to serve as tools for the ITS community.



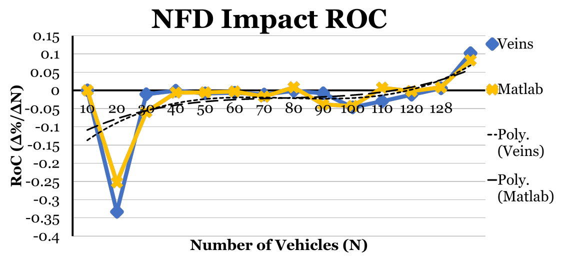
(a)



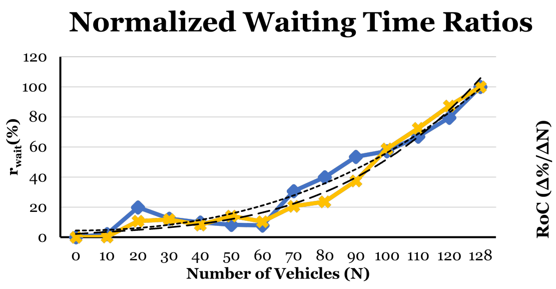
(b)



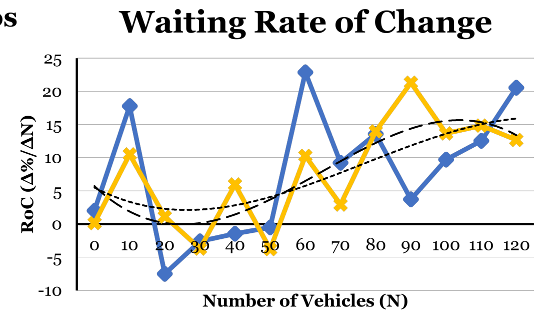
(c)



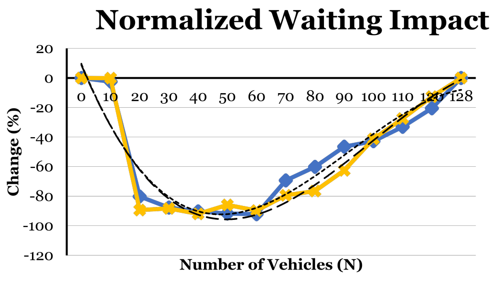
(d)



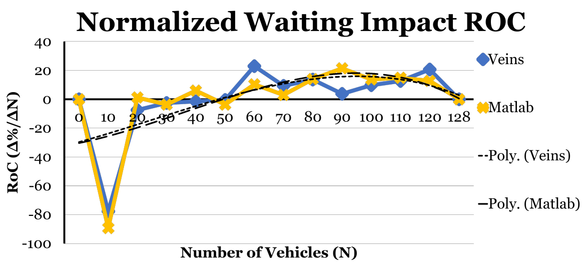
(e)



(f)



(g)



(h)

Figure 4.6: Simulation Results of Both Architectures for ASL and No Attack.

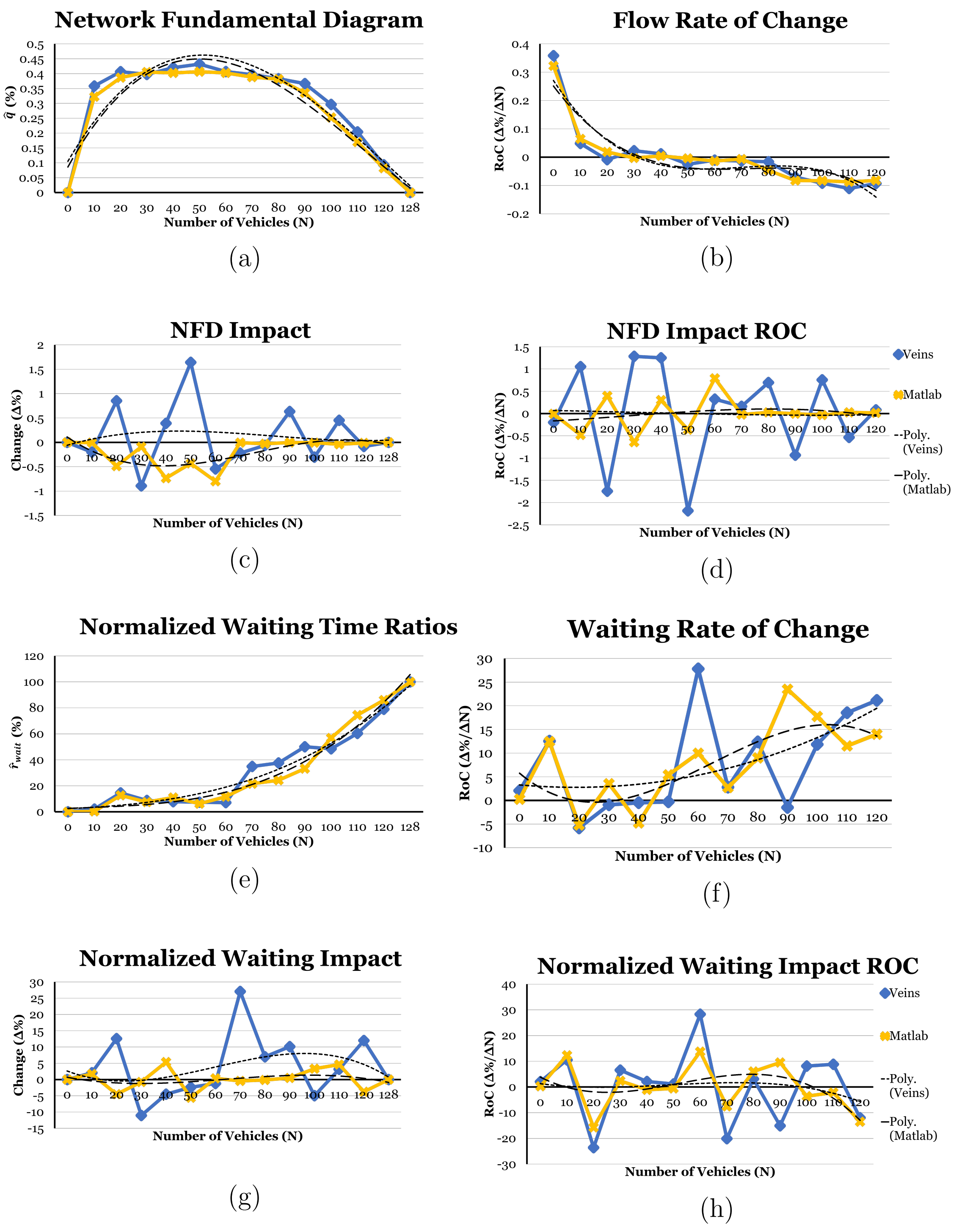
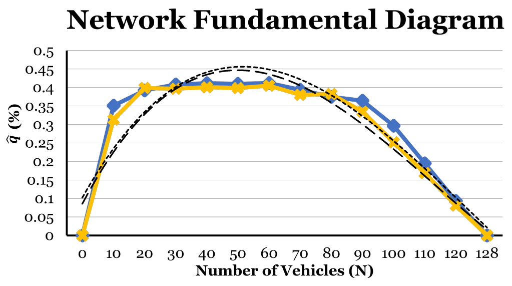
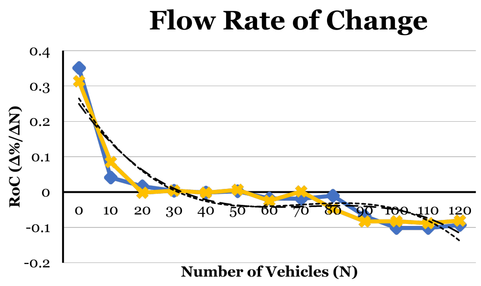


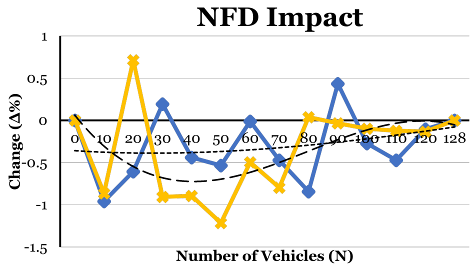
Figure 4.7: Simulation Results of Stealthy Attacker, Random Attack Value $\hat{\alpha}(v(t)))$, and Static Aperiodic Timing.



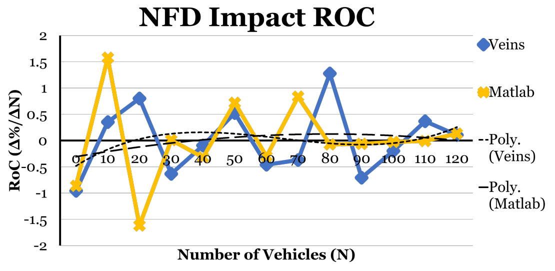
(a)



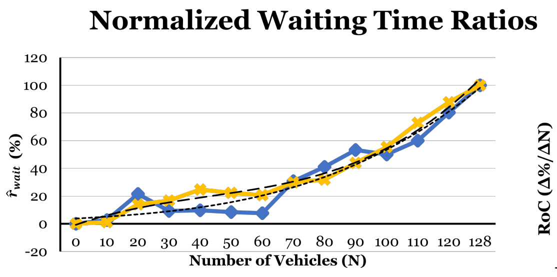
(b)



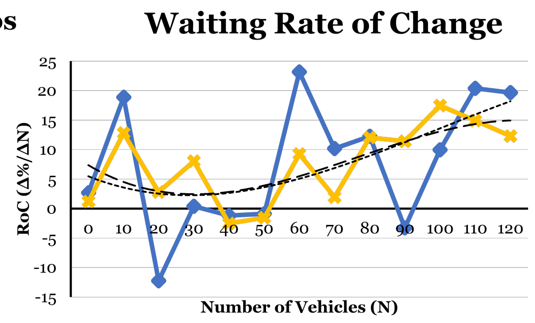
(c)



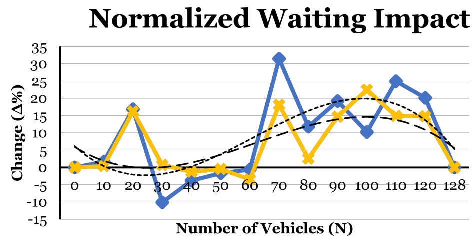
(d)



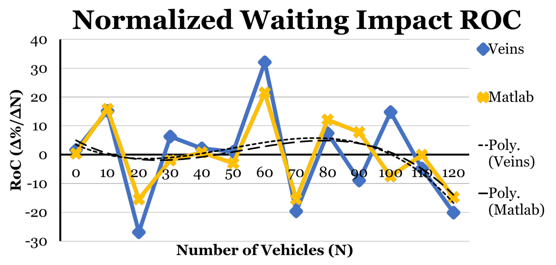
(e)



(f)

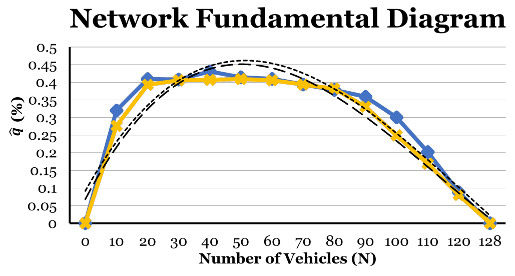


(g)

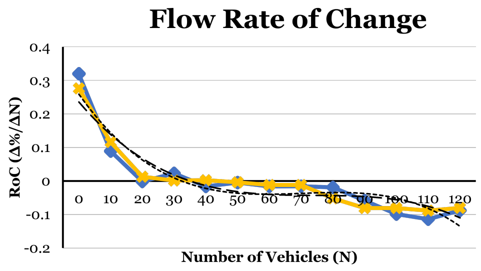


(h)

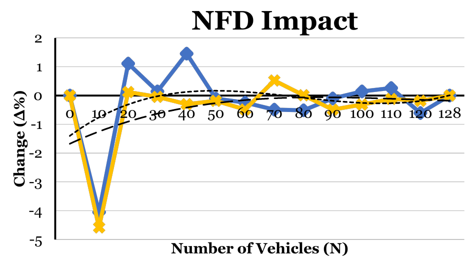
Figure 4.8: Simulation Results of Moderate Attacker, Random Attack Value for $\hat{\alpha}(t_{comm})$, and Static Aperiodic Timing for Entire Simulation, but Random Vehicle Attack Probability $\hat{\phi} = 0.4$.



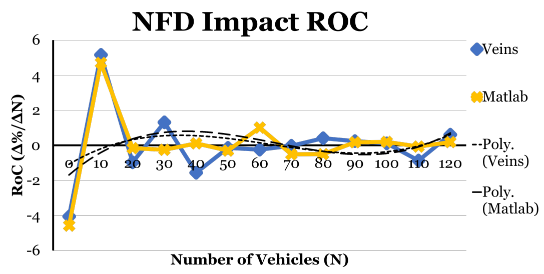
(a)



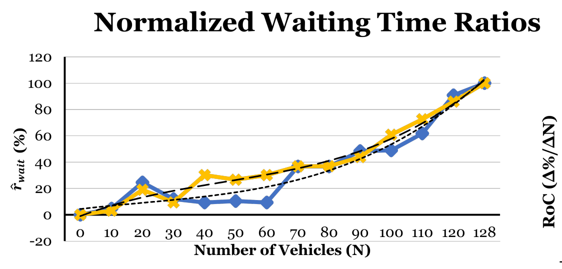
(b)



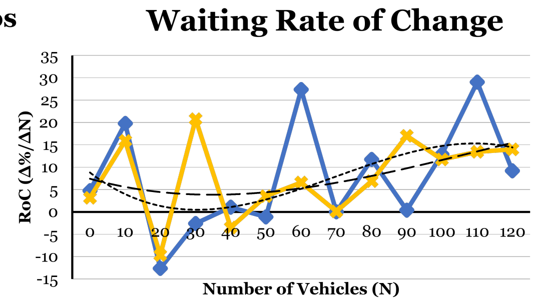
(c)



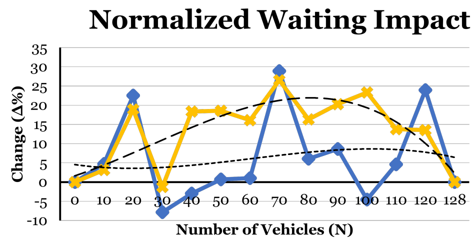
(d)



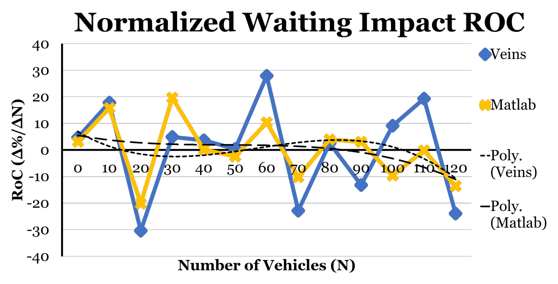
(e)



(f)



(g)



(h)

Figure 4.9: Simulation Results of Moderate Attacker with Static Attack Value $\hat{\alpha} = 60m$, and Static Periodic Timing of Attack Period $\hat{T} = 30s$ and Offset $\hat{\phi} = 30s$.

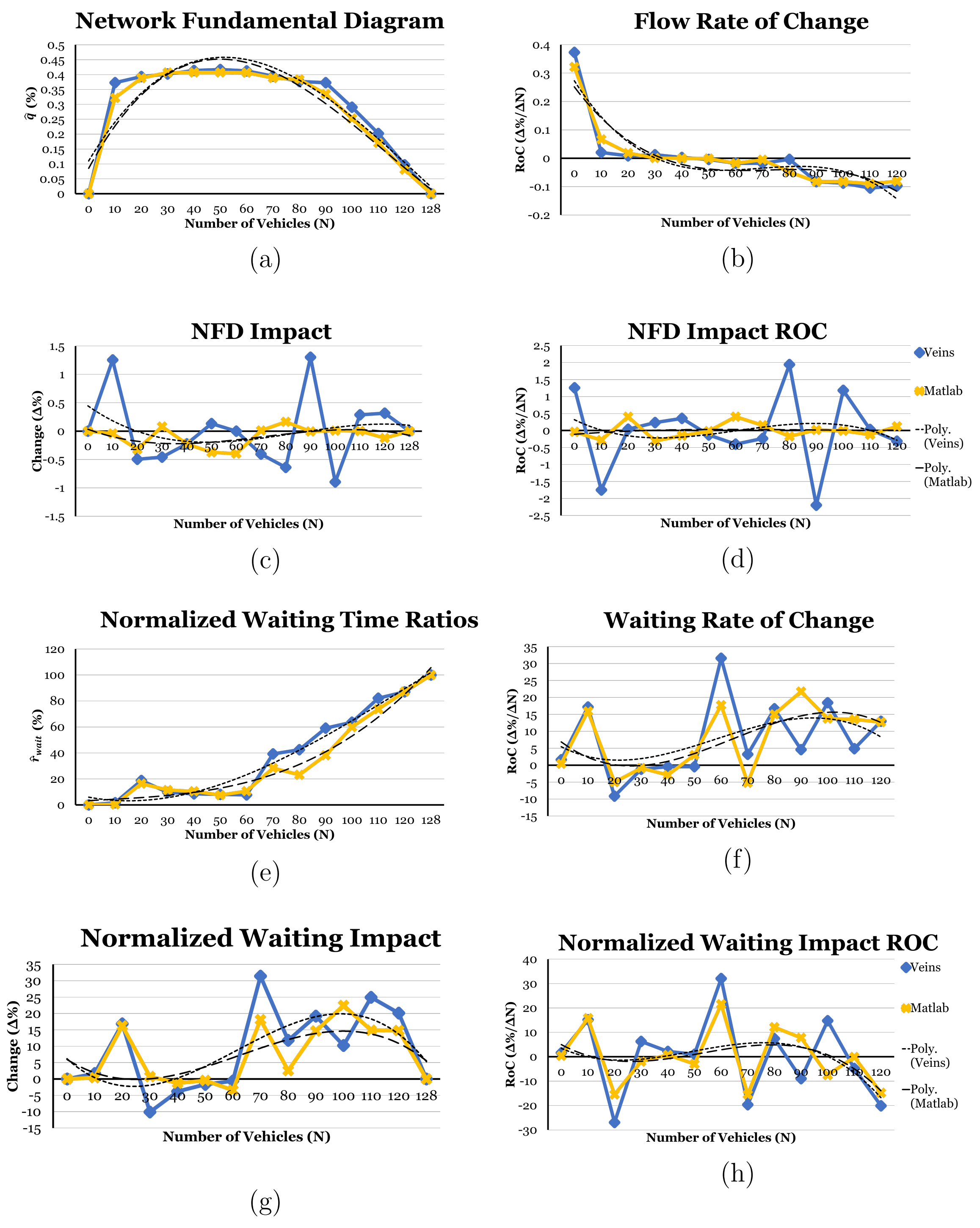
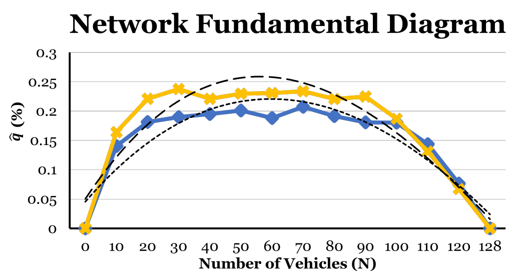
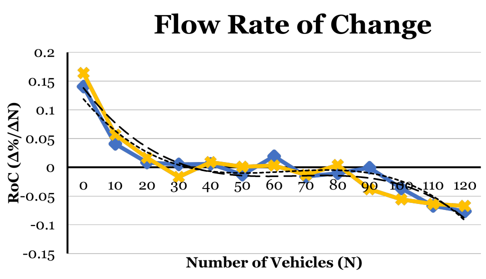


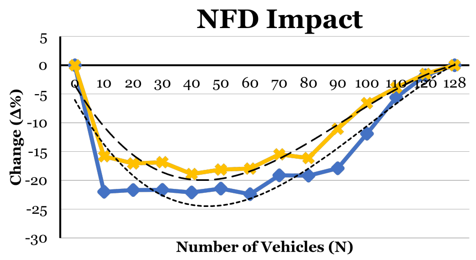
Figure 4.10: Simulation Results of Moderate Attacker, Attack Value $\hat{\alpha}(vif) = -21\ veh$, and Aperiodic Timing.



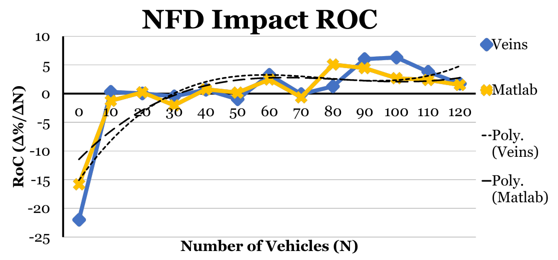
(a)



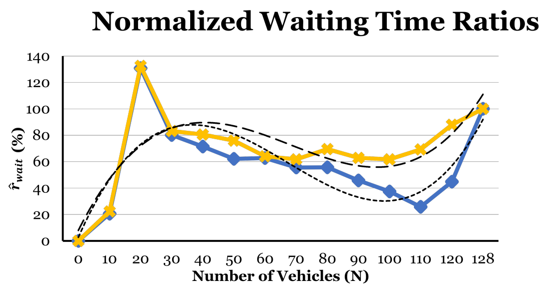
(b)



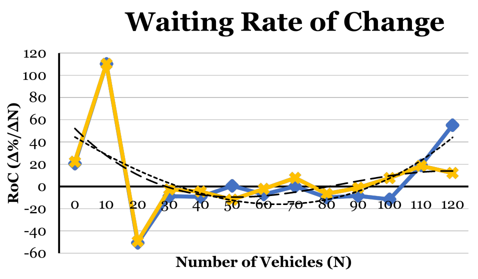
(c)



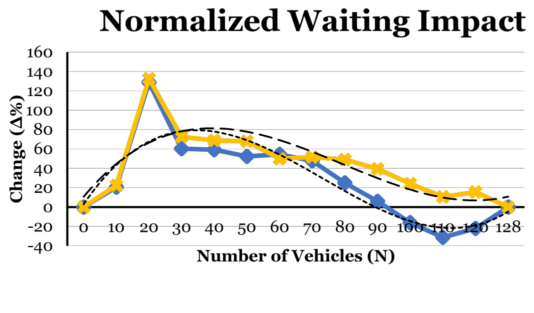
(d)



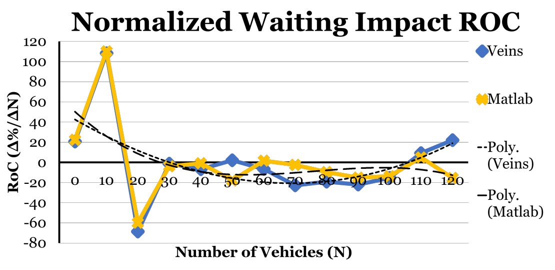
(e)



(f)



(g)



(h)

Figure 4.11: Simulation Results of Extreme Attacker, Random Attack Value for $\hat{\alpha}(x)$, and Random Attack Time Probability $\hat{f}_t = 0.8$, with Random Attacked Vehicle Probability $\hat{f}_n = 0.5$.

a blue line with diamond markers. Notice the similarity between the trend-lines in all the figures. It is interesting to note that ASL acts as a sort of bounded deceleration, which the BA-Newell Model lacks. Hence, although the two car-following models are fundamentally different, the results match up closely when normalized. In contrast, when there is no ASL (not provided), there is up to 30-50% difference in Average Waiting Time Ratios for the region of saturated densities.

The attack modeling metric results (all in percentage) are shown in Figures 4.7 to 4.11 for various types of attacker profiles and attack types. Exhaustive results are not provided because there are many possible attack configurations. It is also important to note that the values should not always be taken at face value; instead, one may infer valuable information from the plotted 3rd degree polynomial trend lines. Through simple observations, the NFD and Waiting graphs (a and e) and their RoC graphs (b and f) are the most similar for the two architectures, and their trend lines match up quite closely. Additionally, although the impact metric graphs (c and g) do not align as well in several cases (mostly due to car-following differences), their RoC counterparts (d and h) do, with average distances across all examples ranging from 1.8-3.5% for for $\hat{RoC}_{q\Delta}$ and 3.3-9.6% for $\hat{RoC}_{wait\Delta}$, and standard deviation ranges, 1.7-3.5% and 3.0-8.7%, respectively. Interestingly, in Figure 4.11, exceptional similarity is observed between the two architectures across the board. Most likely this is due to the attacker model being an extreme one resulting in more prominent effects. Overall, the results support the claim that the trends are transferable despite the fundamental differences in simulation tools, network designs, and car-following models in the architectures.

In addition to the fundamental architectural differences, the discrepancies between the values may be a result of the randomness embedded within the ASL application or within the attack models. There are extreme variations in the range $N \in [20-90]$ (some of the largest impacts occur in graphs c and g within each figure) because this is where the network is the most dynamic and vulnerable (i.e., the overall network behavior may range from complete free-flow to congestion), compared to the network densities on the opposite sides of the spectrum [3]. Further, it is observed that the value of $\hat{\gamma}(var[i], t, n)$ may actually incur an opposite effect on the system than what the attacker would have intended (e.g., a positive $\hat{\gamma}$ may force the system to actually help the vehicle make the green time faster or perhaps the effects will help out highly congested traffic densities more than normal ASL). Thus, to cope with this, one may program the attack model to dynamically change the sign of γ or stop the attack on a vehicle or time step to make the impact desirable for an attacker. For example, when the average density is within a certain range (such as the oversaturated densities where $N >\approx 90$ in Figure 4.7) or when the attack may actually help out the vehicle have less waiting time based on the current phase.

More intelligent attack models may be constructed from the inferences made from these basic examples. For example, attacks that are permutations and combinations of other ones; attacks that smartly make use of the timing and network state (average density, leader's position, etc.); attacks that incorporate budgets and costs, etc. As the V2X ASL is an ITS use case, other use cases dedicated to improve average speed and average waiting time may also be subject to similar adverse impacts. On the other hand, inferences to improve the security of the system may be made as well. For example, note that the saturated densities ($N \in [10, 90]$ in Figure 4.7) appear to be the most vulnerable. Hence, more resources may be dedicated to protect the ITS when its current average density is in this range. Besides the ITS use case design implementation, other parameters such as aggressiveness, connectivity, and sensor locations may all be studied to improve an ITS' security level as well. Theoretical and practical security solutions may be implemented and evaluated (in terms of overhead and performance improvement) too. Using the methodology, taxonomy, metrics, and tools, all these aforementioned ideas are made possible, especially for those interested in ITS and not from cyber- and system-security related fields.

[3] The impact metric values for some lower densities are not normalized since they would be divided by a less than 1.0 number and therefore appear to be much different than the rest. Nonetheless, since the trends are of more interest, this does not negatively impact the results.

4.9 Summary

In this chapter, a methodology and taxonomy are presented to model, simulate, and meaningfully evaluate the exposure and impacts of ITS use cases, such as Advisory Speed Limit control, to attacks. Two distinct architectures and tools are constructed to simulate the models and evaluate the metrics. It was observed that there is consistency and transferability across the results. The methodology may serve as a framework for more intricate and interesting attacks to evaluate the security of an ITS design. For example, it may be useful for integrating security regarding other challenging components of ITS such as networking or IoT. One such integral subcomponent of IoT devices that is often overlooked in security is the battery system. For this reason, the next chapter will tackle this challenge in more detail and discuss battery system security within the IoT of ITS.

Chapter 5

Battery System Security in the Internet of Things - An ITS Perspective

5.1 Introduction

In our economy, all types of batteries play important roles to help drive various types of systems that are part of the Internet of Things (IoT). The IoT includes systems that are interconnected with each other and their environments via software, hardware, sensors, actuators, and network connectivity. Some examples of IoT include cyber-physical systems (CPS) and mobile systems. Mission-critical CPS used in transportation, manufacturing, power grid, military and more all require batteries with high energy density and power density to ensure long-time safety and functionality. Mobile systems primarily require small, yet high energy density batteries that do not easily lose capacity for the satisfaction of consumers. The Li-Ion battery is one type of battery that fits these conditions and is garnering a huge amount of attention. According to the Department of Energy (DOE), by 2020, the global Li-Ion battery market is expected to quadruple [72]. It is even predicted that by 2024, the market for Electrical Vehicles will increase up to approximately \$270 billion with an average price of \$30 thousand. Tesla alone is expected to consume over 2 billion highly efficient Li-Ion cells by the end of 2017 [144]. It will be a significant challenge in the near future to ensure that these and other types of batteries are trustworthy and functional.

These batteries are generally prone to thermal runway as a result of improper charging/discharging procedures, of defective materials, and/or of environmental effects. As battery manufacturers aim to pack more energy into smaller batteries, the risk and danger of using them increase. For this reason, safety circuits and battery management systems play particularly crucial roles in preventing such explosions. In addition to safety, however, certain security requirements must be guaranteed to users of battery-operated systems. These requirements include *confidentiality, integrity, availability*, and *authenticity* [111, 25]. In the case that an unexpected attack occurs, the system should also have detection, recovery, and resilience methods. Often, the battery is overlooked in the security analysis of these systems, but rather looked at in other types of system analysis (e.g., efficiency). This section addresses the lack of battery security analysis works by identifying and evaluating different attack vectors from different system layers. Attacks that initiate from one layer and affect other layers are called "cross-layer attacks", as coined in [22]. The battery system is detailed in Figure 5.1, where it is abstracted with three layers: the **Application Layer**, the **Battery Management System (BMS) Layer**, and the **Physical Layer**. Since these layers are interconnected, the attackers may develop more sophisticated attack vectors from simpler ones.

The challenges at the physical layer are safety and security ones: *integrity, availability, authenticity.* Given the size of the battery market, it is no surprise that there are battery counterfeiters. Counterfeiters aim to profit from undermining the high quality battery

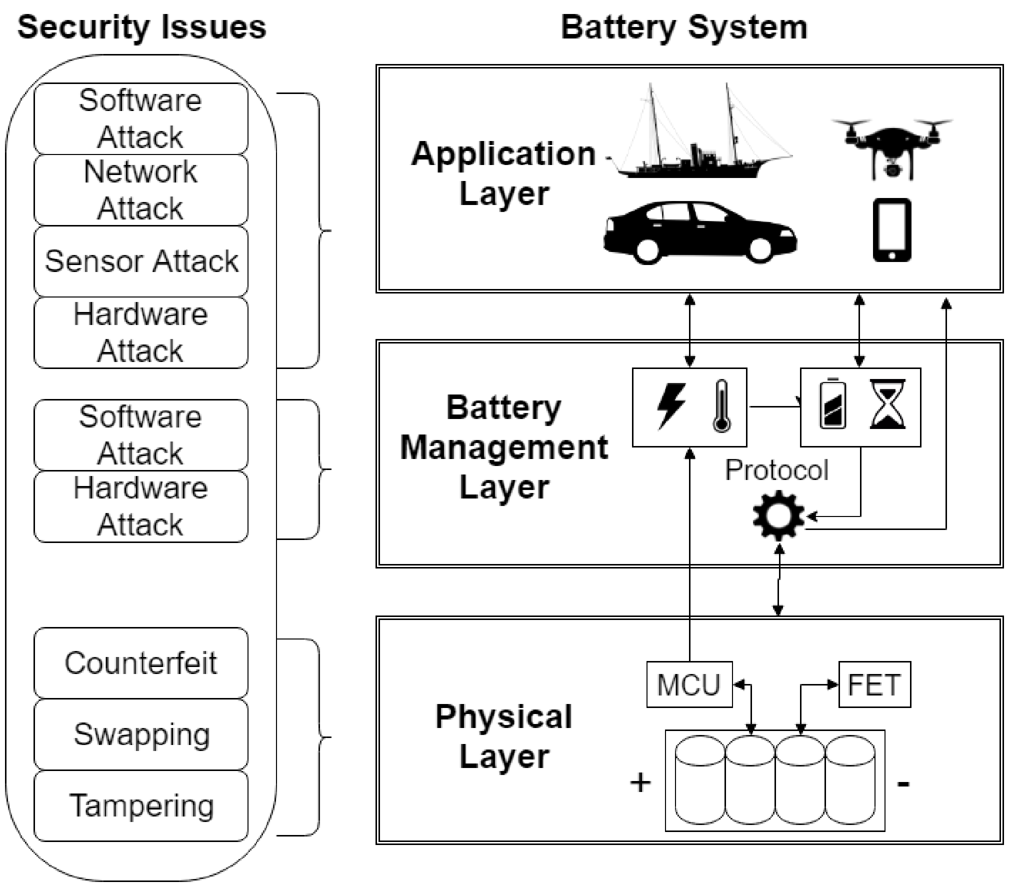


Figure 5.1: Overview on battery system and security issues.

market by creating their own low-quality batteries or reusing older ones. In addition to counterfeit, there exist replacement/swapping and tampering attacks that also breach the security requirements. One layer up is the BMS layer, which ranges from a simple subsystem (only sensors) to a complex one (with sensors, models, and learning techniques). The BMS is able to estimate and predict the battery state to make decisions for functionality, efficiency, and safety. Unfortunately, potential security risks come with such features. If any malicious entity is able to gain direct or indirect control over a BMS, they will have the power to weaken or damage the overall system by controlling the battery-related protocols. The security concerns here correspond to *availability* and *integrity*. Lastly, the application layer may prove to be another avenue for attackers to gain access to the battery system. Cyber-Physical Systems (CPSs) and mobile systems are used as case studies for this layer and the potential security risks of attackers leveraging software to directly or indirectly negatively affect the battery are discussed. The application layer of the battery system is susceptible to security attacks which affect the *availability*, *integrity*, and *confidentiality* of the overall system.

This section describes and evaluates the security issues of battery systems in IoT and is organized as follows: **Subsection 2** will discuss the potential attack vectors on the battery system within the physical layer, the BMS layer, and the application layer. **Subsection 3** will discuss approaches and challenges of related works on battery system security and safety. Finally, **Subsection 4** discusses a proposed solution for some of the battery security issues discussed in this chapter.

5.2 Battery System Security Issues

5.2.1 Physical Layer

The physical layer of a battery system includes the battery cells, the surrounding circuitry, and the connections with the Battery Management System (BMS). The battery cells have certain limits (lower and upper voltage/current bounds) and demonstrate specific behavior towards different power requests. The BMS and the circuity components (e.g., fuses) are responsible for monitoring the battery cells and protecting them from overvoltage, undervoltage, overcurrent, overloading, and also overheating. See **Figure 5.2** for an abstracted model of the physical layer of a battery system.

The behavior of the battery cells may be modeled and described using an equivalent electric circuit model [191, 236, 280]; the battery cell is modeled as a variable-voltage power supply in series with an internal resistance. The ratio of the available charge to the battery capacity is represented by State-of-Charge (SoC) which changes over time as the battery is under

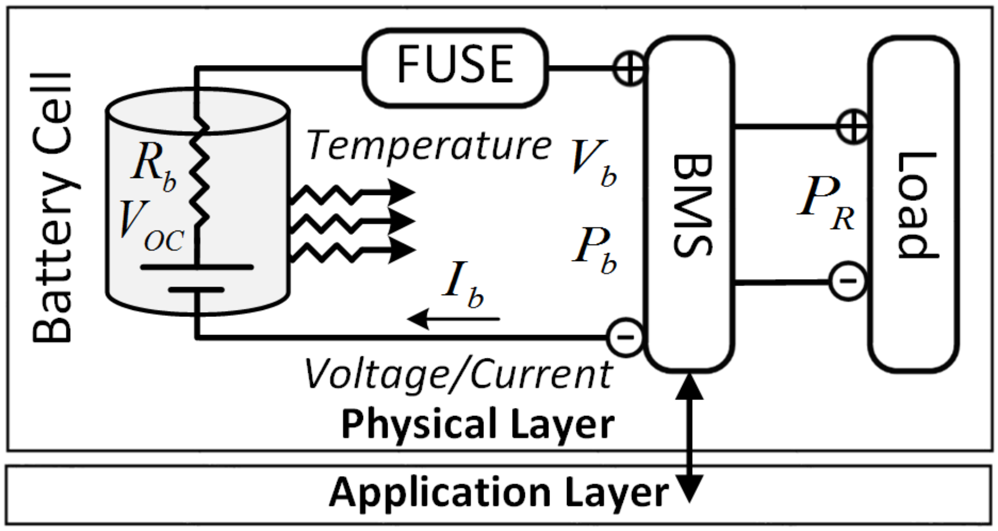


Figure 5.2: Physical layer of battery system.

utilization. The open-circuit voltage (V_{OC}) of the battery (the variable voltage power supply) and the battery internal resistance (R_b) depend on the SoC value. Therefore, the current going through the battery (I_b) and the terminal voltage of the battery (V_b) significantly depends on the battery power (P_b) and the battery parameters. Moreover, the battery cells generate internal heat while charging or discharging and changes the battery temperature in a positive feedback manner. The heat generated is caused by the power loss due to internal resistance or the entropy change in the ions [145, 236]. The amount of the generated heat is also dependent on the battery power and other parameters. It needs to be noted that the environment conditions such as ambient temperature and packaging heat dissipation factor influence this behavior as well.

Typically, the BMS implemented on a microcontroller unit utilizes the sensors connected to the battery cells in order to gather the values of the above-mentioned variables (e.g., current, voltage, and temperature). These values are then filtered out to remove the noise, for instance by using a Kalman Filter [119], and to estimate the battery state and prevent safety-threatening operations. However, the overall safe operation of the BMS depends on having the right knowledge of the battery parameters. For instance, an unsafe operation may occur if the battery parameters such as control limits and modeling variables are different or get changed from what the BMS knows. This may happen by any attack on the battery or BMS, which will be explained further.

5.2.2 Attack Model

The effects of supplying a counterfeit battery into a system may severely affect the *availability*, *integrity* and *authenticity* of the system. It may also result in economic costs for both battery system manufacturers (due to warranty claims) and consumers (repairing/resupplying) [76]. This is due to not only a possible lack of safety circuitry in battery packs but also the cheap materials and manufacturing process used in making the counterfeit batteries [93, 165]. Counterfeiters also resell degraded batteries that are dangerous to use and more difficult to detect. The potential cost of using counterfeit batteries rises with lower quality authentication schemes (e.g., none, form factors, barcode, radio frequency identification), while the cost of preventing counterfeit batteries rises with higher quality authentication schemes (e.g., hashing, cryptography) [273]. **Figure 5.3** depicts the counterfeit security issues.

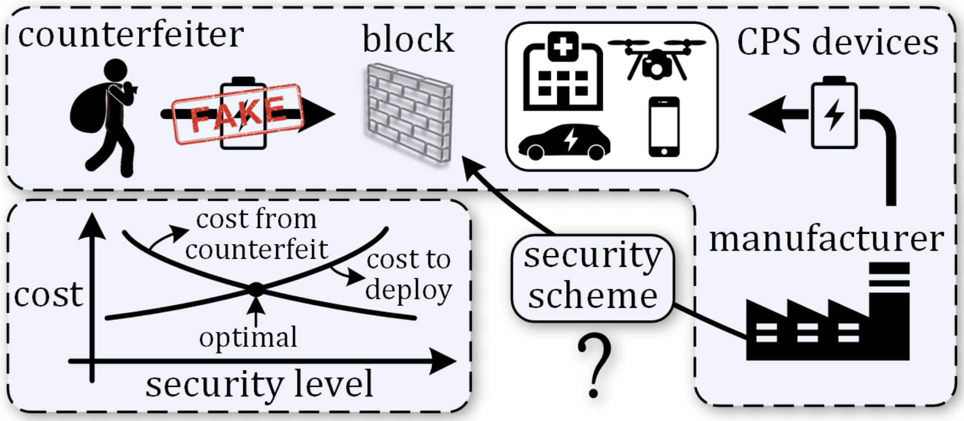


Figure 5.3: Counterfeit security issues.

Generally, a battery is manufactured and sent to the CPS manufacturer through shipping by third-party distributors. However, the received batteries on the CPS manufacturer side may not be authentic because a man-in-the-middle attacker replaced them, or the manufacturer deliberately sent fake batteries. The attack model involves an attacker who wishes to modify the authentic battery or replace it with counterfeit at any stage in the supply chain, to either profit from selling cheaper batteries, or to intentionally put others at risk. A real-case scenario occurred when a Simi Valley CEO sold counterfeit batteries worth more than $2.6 million to the U.S. Department of Defense, who used them in the critical systems on submarines and aircraft carriers [270]. Because the batteries had been covered with counterfeit labels of the approved manufacturers and had spoofed form factors, they were used before eventually being detected.

In another example, in 2009, the customs authorities from an airport in Germany found counterfeit mobile phone batteries branded as Siemens, despite Siemens stopping its mobile phone business many years ago. These batteries turned out to have no protection circuitry and were easily ignitable if they came into contact with water [76]. There have been several incidents that indicate the risks of using either counterfeit or faulty batteries: The Samsung Galaxy 7 case [65]; cellphones bursting into flames [156, 265]; battery explosions in self-balancing scooters known as "hoverboards" [195]; Li-Ion batteries catching on fire in Boeing 787 Dreamliner [198]; and a Tesla car catching on fire within just 5 minutes of being used [20]. In some of these incidents, the sources state that it is unknown whether or not the batteries were counterfeit. This provides additional motivation to determine the authenticity of batteries to prevent such scenarios.

Another attack may occur during offline or runtime (when the user is using a battery-operated system) where an attacker could tamper with the safety circuitry on the battery pack and/or replace existing batteries with lower-quality/faulty ones to cause system failures. An example would be replacing batteries in systems to cause inefficiency, missed deadlines or safety risks. An attacker could even replace backup batteries used for emergency situations (e.g., during power outage). In one use-case, researchers on Unmanned Aerial Vehicles (UAVs) are looking toward automated battery replacement to expand UAV operation in areas too risky for humans [255]. However, with less human-in-the-loop interaction in the battery supply chain, there is a potential vulnerability where enemies or other attackers may replace batteries at the battery supply location. In another case, if attackers may capture a UAV by making it appear that it fell down due to a glitch (such as in the Iran-U.S. RQ-170 incident in 2011 [205, 90]) they may be able to replace the UAV's battery or tamper with its battery pack circuit before releasing it again. The battery swapping threat also exists for the Electrical Vehicle and Electric Scooter. Electrical Vehicle (Tesla) and Scooter (Gogoro) manufacturers are planning to create battery swapping stations to help ease range anxiety issues for users [287]. On top of rechargeable battery systems, it needs to be noted that the threat model also includes battery swapping for longer-lasting non-rechargeable battery systems in IoT devices. Such a threat exists if one assumes that the attacker has a sufficient level of physical access to the battery during system runtime. The attacker may then be able to replace an IoT device's battery with a shorter-lasting one or a defective one. As these systems are generally expected to last a long time to ensure a high level of availability for many systems, the battery swapping attack could cause a major breach in availability and cause a "domino effect" on the availability of other dependent systems. It is clear that if a legitimate battery was swapped, the security of the system and the safety of the user may be seriously threatened. One may generalize counterfeit, replacement, and tampering as a single attack model, as shown in **Figure 5.4**.

5.2.3 Battery Management System Layer

The BMS may be any system that manages the battery [19]. As discussed, overcharging and deep discharging may damage batteries by shortening their lifetime and even cause more hazardous situations. This requires the adoption of a proper BMS to maintain the states

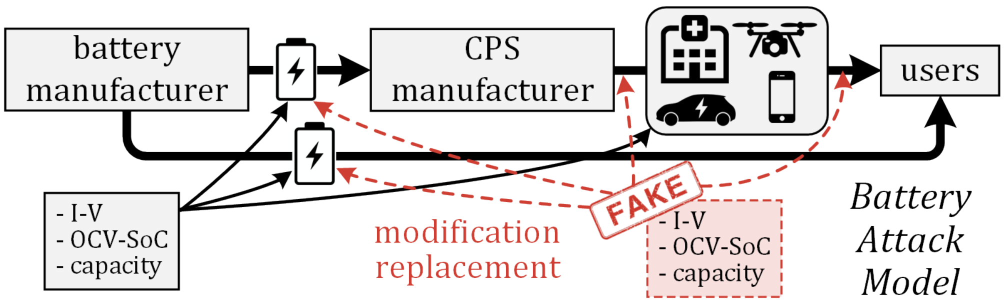


Figure 5.4: Abstract attack model on batteries.

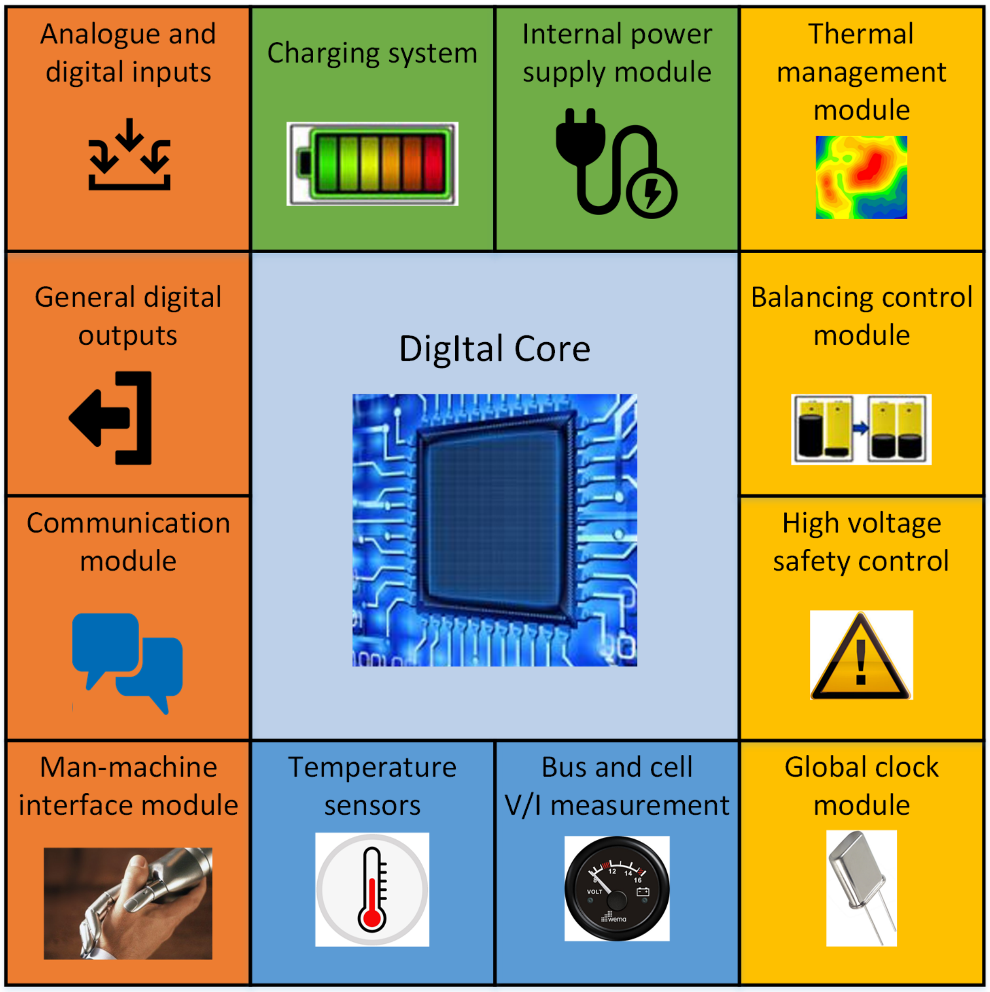

Analogue and digital inputs
Charging system
Internal power supply module
Thermal management module
General digital outputs
DigItal Core
Balancing control module
Communication module
High voltage safety control
Man-machine interface module
Temperature sensors
Bus and cell V/I measurement
Global clock module

Figure 5.5: Framework of software and hardware of BMS.

of each cell of the battery within its safe and reliable operating range. In addition to its primary functionality of battery protection, a BMS should estimate the battery status in order to predict the actual amount of energy that may still be delivered to the load [43].

The battery could be a single cell, battery module, or battery pack and could be rechargeable or non-rechargeable. The system could manage the battery by monitoring the battery, estimating the battery state, protecting the battery, reporting the data, balancing it, etc [178]. A BMS may include any of the following functions [19]: monitoring; protecting; estimating; maximizing; and/or reporting the battery state to users and/or external devices.

5.2.4 BMS Functions

Battery management is mandatory for Li-ion batteries to ensure energy availability and lifetime, and the safety of the energy storage system. To do these, a BMS must at least do the following [19]: Prevent overvoltage/undervoltage; prevent overheating; prevent low temperatures; and prevent the overcharging/undercharging. The basic framework of software and hardware in the BMS is shown in **Figure 5.5** [178]. The BMS would have inputs such as: a main circuit current sensor and a voltage sensor to measure the main current and voltage; temperature sensors to measure the temperature of the cells, the temperature outside the battery box,and maybe also the temperature at the battery coolant inlet and outlet; general analog inputs from sensors of specific applications; and general digital inputs like charging allowed/banned, etc.

The BMS would have outputs to modules such as: a thermal management module (including a fan and/or electric heater); a balancing module (including a capacitor with switch array and dissipation resistance) to do the battery equalization; voltage safety management (including a main circuit contactor and battery module contactor); general digital outputs (e.g., display of battery status, charging indicator, failure alarm); and a communication module. Also the BMS would have the internal power supply module and global clock module, and it may have the charging system and man-machine interface module.

5.2.5 Attack Model

The BMS is a combinational structure of hardware and software that interacts with the application layer, and affects and controls the battery system's physical layer with state estimation/prediction, parameter detection, safety control, charge control, battery equalization, information storage and so on. It makes a BMS a powerful tool, which if vulnerable, could be used by adversaries to attain valuable information and control all BMS functions, including ignoring critical battery conditions and tolerating high voltages and currents, to damage the battery and even to ignite a fire [229]. Actually, without proper protection on the safety and security of the BMS, the more functional a BMS is, the more vulnerable a battery system could be, as the adversary may manipulate more behavior of the battery and get more information.

The BMS hardware is comprised of sensors, actuators, and controllers. Integrated Circuit (IC) counterfeiting and/or tampering may happen in many phases of the BMS supply chain, including IC manufacturing, system manufacturing and integration. They may affect the availability, integrity and authenticity of the battery system in different ways. For instance, degraded ICs that are recycled, remarked, out-of-spec or defective will not only lead to economic loss but also cause functional downgrading and even system damage [105]. Tampering may either be on the die level ("hardware Trojan") or package level. The malicious or affected circuitry due to tampering may act as a silicon time bomb where the BMS will start behaving differently (such as draining or aging the battery) under certain conditions, or act as a backdoor where secret information from the system may be sent out to an adversary.

The BMS software is comprised of the programs in the controllers and firmware in all the ICs. The software of a BMS may be tampered either during IC manufacturing or system integration. Given simplistic security measures such as a deterministic password, an attacker may gain direct access to the BMS's firmware [187]. Moreover, cross-layer attacks initiated from the application layer may affect the BMS layer, which makes it possible to replace the BMS software remotely through Bluetooth, Wi-Fi, etc. In attacks targeting availability, malicious software could be injected to change the normal behavior of battery and control the charging, equalization of the battery, etc. In attacks targeting confidentiality, critical data stored in the BMS, such as SOC, SOH, accumulated charge and so on, may leak to the adversary due to malware inserted in BMS. Attackers may leverage the BMS functionalities to conduct more sophisticated attacks, as shown in the following section.

5.2.6 Application Layer - An ITS Perspective

Attack Classification In battery security, the attacks may be broadly classified based on the security objectives and action characteristics. Similarly, the mobile battery and cyber-physical system attacks via the application layer may be classified into three categories depending on the adversary's intention to damage or disrupt availability, integrity, and confidentiality of the critical functions and information within the system. The various types of battery attacks include overcharging, draining, information leaking, and illegally modifying user sensitive information exchanged by data network, mobile and cyber-physical system applications, and the battery management system.

- **Attacks on Availability:** Attacks targeting availability are also called denial of service attacks. The action characteristics of these attacks include attempts to disrupt the battery service availability of the system which may lead to detrimental effects on the system.
- **Attacks on Integrity:** The action characteristics of attacks on integrity include deliberate attempts to modify or disrupt the device battery functionality or information exchanged by data networks, battery management systems, and the system.

- **Attacks on Confidentiality:** This kind of attacks may be defined as the attempts to leak unauthorized information about the user or the overall system through attacks on the battery system.

5.2.7 Safety Critical Cyber-Physical System Applications - Electric Vehicle (EV) as Use-Case

This section takes a look at the security of ITS with respect to battery-operated subsystems, particularly EVs. These systems are constantly in motion and have processing capabilities for highly interacting with the physical environment in a feedback manner to control certain functions. They highly depend on batteries to perform their autonomous, mission-critical, and/or safety-critical functions. These systems require high quality batteries with strict constraints, such as high energy density, and accurate BMSs to ensure that the batteries will not fail or cause catastrophe in their runtime environments, which could lead to financial loss, physical damage, and even human injuries or casualties. Some examples of these systems include Electrical Vehicles (EVs), Solar-Powered Management Systems (SPMS), Unmanned Aerial Vehicles (UAV) and Autonomous Underwater Vehicles (AUV), and spacecraft systems. Because each system is uniquely designed for different application purposes, it is apparent that they have both unique and common battery system security challenges.

Electric Vehicles (EVs) carry a large amount of batteries packed together to sustain varying demands over time and to guarantee a sufficient driving range for consumers [278, 279]. An EV may have BMSs at the cell, the module, and the pack levels. These BMSs are then connected with each other via an internal may bus and with external electronic control units (ECUs) of the vehicle via an external may bus. ECUs are connected with each other via different intra-network protocols and technologies (e.g., MOST, LIN, CAN, FlexRay and Ethernet designed for various objectives (e.g., safety, efficiency, etc).

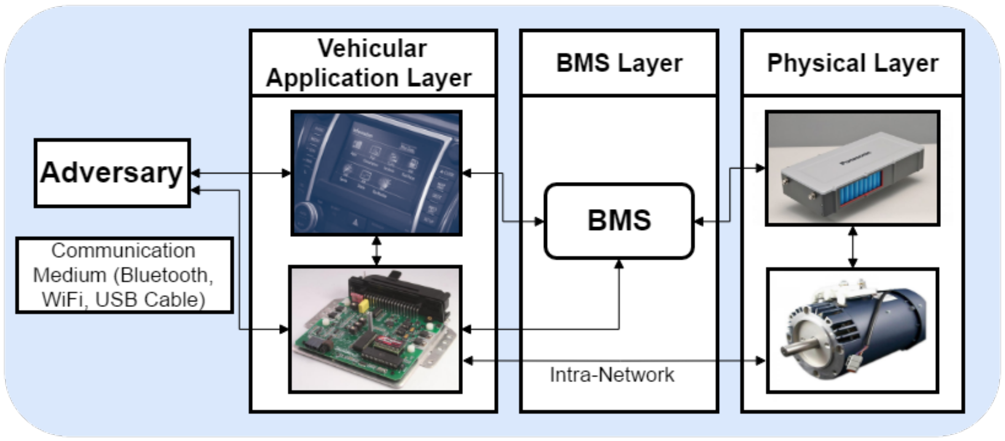

Vehicular Application Layer
BMS Layer
Physical Layer
Adversary
Communication Medium (Bluetooth, WiFi, USB Cable)
BMS
Intra-Network

Figure 5.6: Abstracted battery system attack model on electrical vehicle.

As discussed in Chapter 2, with access to the intra-network, it turns out that it is highly viable for an attacker to exploit and conduct a wide variety of attacks on the automotive system. In addition to those aforementioned attacks, they may perform most of the BMS-layer attacks (discussed more in detail in Section 5.2.3). The intra-network may be accessible to attackers via malware or malicious websites that applications may stumble upon through different communication mediums, such as telematics, Dynamic Short Range Communication (DSRC) used in vehicle to vehicle or vehicle to infrastructure communication (V2X), Bluetooth, Wi-Fi, and USB cable [53]. However, it may be possible to implement indirect attacks on the battery system via unique application layer features, such as by disabling the regenerative braking system or by altering the Heating-Ventilation-Air-Conditioning (HVAC) system without the passenger noticing. These attacks end up draining the batteries and therefore affect their availability, and in turn, affect the EV availability to the users. Another sophisticated attack may include draining the EV battery by spoofing sensor information sent to ECUs via the intra-network [130] or by future V2X features such as data transfer for entertainment or personal use. Sensor information is crucial to the efficiency and safety guarantees that EVs should provide [15, 281]. Examples of sensor information are: the GPS location, proximity data, path information, velocity and relative velocity, and tire pressure.

EVs can also potentially present greater risks than combustion vehicles during accidents as shown by NHTSA crash tests in 2011, in which the Chevrolet Volt EV caught on fire twice, prompting the NHTSA to open an investigation into the vehicle's fire risk [73]. Furthermore, the widespread adoption of rapid charging technology (such as Tesla Superchargers) places even greater stress on EV batteries, requiring active cooling, dynamic charge rate based on cell temperature, and robust cell-balancing to ensure safe rapid charging. However, in the case that these safety measures fail, the results can be disastrous. For example, in two separate incidents in 2016 and 2019, a Tesla vehicle caught fire while plugged into a Supercharger station [162, 163]. Both fires were attributed to short circuits in the vehicles' electrical systems.

The aforementioned examples demonstrate that EV battery technology's complex architecture and high risk-factor present a large attack surface. The high impact of battery failures makes this subsystem an attractive target for attackers and presents a significant risk to EV owners. Since the battery management system is usually connected to the CAN-bus and several groups have already demonstrated cyber-physical attacks on EVs [5, 6, 122, 8, 149], it is only a matter of time before exploits targeting vulnerabilities of the battery subsystem are revealed.

Attacks Targeting Availability Lithium-ion batteries have various failure modes ranging from reduced battery life/performance to complete battery failure and thermal runaway. The former failure modes can be triggered via excessive cell cycling, charging past 100% capacity, or malicious tampering of vehicle loads such as manipulating HVAC settings, disabling regenerative braking, or disabling the discharge limiter to deep discharge the battery. Complex, new EV control systems that use machine learning and artificial intelligence to improve efficiency such as that proposed by Lin et al. [172] are potential attack vectors to manipulate the battery subsystem and drivetrain of the vehicle. Several groups have shown that machine learning models are highly vulnerable to adversarial attacks [159, 80], meaning that machine learning-based control systems can potentially be leveraged to attack the battery subsystem and result in these failure modes.

Thermal runaway can be induced via a combination of factors including a high charging rate, poor cooling system performance, and internal or external short circuits. This failure mode is most likely to occur with rapid-charging devices as the high power output of the chargers increases battery cell temperatures significantly and requires complex thermal management in the vehicle. Some fast-charging systems require vehicles to run active cooling systems while charging to ensure battery temperatures do not reach critical levels. Despite these safety measures, the commands controlling charge rate and active cooling are usually sent via in-vehicle networks, such as CAN.

In Section 2.5, there were various depictions of how attackers can gain access to in-vehicle networks; in this scenario, an attacker with access to the network could potentially manipulate bus traffic to induce thermal runaway (potentially less difficult to detect than other manipulations). Although many battery subsystems have physical safety measures to prevent thermal runaway such as thermally-triggered fuses, these measures are usually irreversible, meaning attacks that induce these conditions can cause permanent damage to a battery pack and compromise its availability.

In addition to vehicle battery packs, battery subsystems are prevalent in sensors, mobile devices, and future V2IoT devices. In general, battery subsystems consist of three layers: application, battery management, and physical. Per each layer, the attack vectors used to gain access may vary and actual exploits may target the confidentiality, integrity/authentication, and availability of the battery subsystem and/or other connected subsystems [177]. Due to the need for low-cost production, battery subsystems tend to be lacking in security across all three layers [67].

At the application layer, attack vectors for battery subsystem attacks involve wireless communication (e.g., vehicular, remote battery management [258]), sensors, telematics, infotainment, EV charging station cables, wireless charging [181], and in-vehicle network ports. The primary attack vector for the battery management and physical layers is the automotive battery supply chain, which consists of many steps that are prone to various exploits (eg, manufacturing, transportation, swapping, recycling). Vulnerabilities in the latter layers include weak software security/hardware, leading to access to the battery management software or the battery circuit.

Attacks Targeting Integrity and Confidentiality Attacks that target the integrity and confidentiality of a system may also be done via the application layer of battery-operated CPSs. These attacks may focus on stealthily forging data of the battery to confuse the system or the user, or by obtaining critical information about the system or the user via the behavior of the battery. For example, supposing an attacker has access to the history of the EV battery usage (achievable by having access to the On-Board Unit or the BMS via the intra-network), the attacker may extract critical information such as the driver's habits and location. However, with knowledge of the habits and location of a driver, an attacker may eventually conduct a bigger breach in the driver's privacy. On the other hand, an attacker may use the access of the may bus of the EV to do an integrity attack by displaying incorrect information about the battery state (e.g., State of Charge (SOC) and State of Health (SOH)) to the user. As a result, the consequences for this attack could be that the attacker unknowingly damages the vehicle or ends up stranded in the middle of a trip with a discharged or unusable battery.

Attacks on confidentiality (via probes or in-vehicle network-based attacks) typically record data related to battery usage to infer user behavior patterns or user location information. Integrity/authentication exploits utilize attack vectors, such as the CAN-bus, to modify charging/discharging protocols (eg, replay, spoofing, message tampering, battery circuit tampering) to disturb battery functionality and/or the functionality of battery-dependent components [106]. Finally, availability exploits (via network-based attacks or battery circuit tampering) attempt to reduce or cut off energy provision to the components needing it [140, 84].

5.2.8 Mobile Applications

The omnipresence of mobile data services and applications expose mobile device batteries to novel security risks. The attackers may exploit unique vulnerabilities in mobile networks, applications, device resources, and network interconnectivity. As battery security challenges mostly arise from cyber-physical attacks launched in conjunction with malicious applications, it is essential to comprehend the potential risks of batteries emanating from the application layer. In general, the attacks via mobile applications are orchestrated by exploiting multiple layers of a mobile device and the communication network. An illustration of possible attacks on mobile battery spanning over multiple layers of communication network, mobile applications, and device resources is depicted by **Figure 5.7.** For instance, if the adversary launches a benign looking malware that is downloaded by the users unknowingly via Wi-Fi, Bluetooth or the other mediums, the malware may deliberately and secretively exploit the battery power (deemed as primary resource in context of battery security) or another resource like a processor, storage, camera, microphone, GPS, accelerometer etc. to disrupt the proper functionality of the device's battery system.

Taxonomy of Battery System Attacks Figure 5.8 is provided to summarize the various types of attacks on battery systems that were discussed. Each attack may target one or more layers and has a set of action characteristics that are manifested into the system. An adversary may exploit one or more of the following mediums to perform their attacks: physical access; sensors; software/hardware; and communication channels.

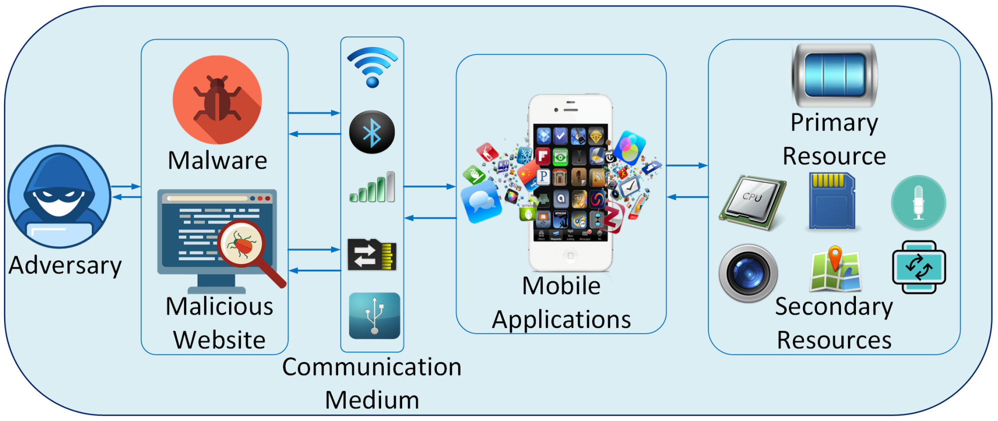


Figure 5.7: Abstracted battery system attack model on mobile devices.

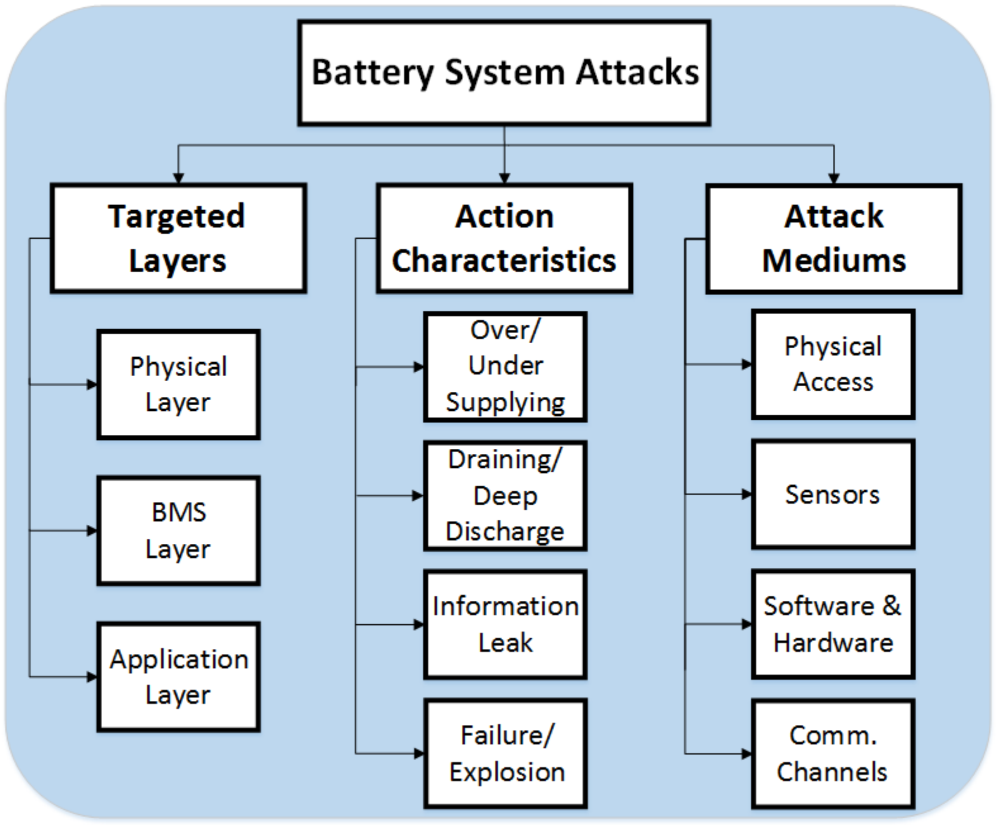


Figure 5.8: Taxonomy of battery system attacks.

5.3 Existing and Proposed Solutions

5.3.1 Existing Solutions

Physical and BMS Layers: Down to the physical layer, there are traditional safety circuits used to prevent the battery or the overall system to go into an unstable state. However, it may be observed that such safety circuits may be tampered with in counterfeit battery packs. For battery counterfeit, some solutions are form factors, barcoding, radio frequency identification (RFID), and hashing/cryptography (e.g., SHA-1/HMAC, KEELOQ, XTEA) [273]. Evidently, the form factors, barcoding, RFID and (some) cryptographic solutions suffer from lack of entropy and therefore are typically easily replicated. On the other hand, although stronger cryptographic solutions may potentially solve these problems, typically their inputs are deterministic (simple keys from the manufacturer of the battery) and therefore may lead to a lack of security [187, 199]. Furthermore, they may add a non-trivial cost to the battery pack [66].

Application Layer Many intrusion detection methodologies have been developed over the years to detect and thwart battery exhaustion DoS attacks on mobile computers and smart phones. Nash et al. [197] developed an Intrusion Detection System (IDS) Framework to mitigate the impact of battery depletion DoS attacks in mobile devices and laptops. Jacoby et al. [131] proposed a Battery-based Intrusion Detection System (B-BID) to prevent the exploitation of battery power via DoS attacks. Moyers et al. [194] developed a hybrid scheme named Multi-Vector Portable Intrusion Detection System (MVP-IDS) that monitors the host-based device instantaneous current (IC) and traffic signatures. The framework recognizes any significant change in the instantaneous current of the device and correlates it to the anomaly or increase in Wi-Fi or Bluetooth traffic

Application layer security should incorporate features to prevent both static and dynamic attacks. To address the static attacks (i.e. the attempts to maliciously modify installed operating system properties and applications) authentication and secure operation verification of executable code and applications is required. As for dynamic attacks that attempt to maliciously inject malwares or run contents from an insecure source, the security methods should be capable of detecting anomalous deviation in operation, power consumption, communication patterns, etc. from the system software's typical executing path of normal behavior. Sensor attacks may be preventable with more reliable sensor fusion techniques and anti-spoofing methods [112]. Following this concept, the next section will detail a proposed solution that takes physical characteristics of the batteries and converts them into a unique fingerprint for secure authentication.

5.4 A Proposed Physics-Based Battery Fingerprint and Authentication Scheme for ITS Devices

5.4.1 Introduction

Existing solutions for physical battery safety typically deal with identifying or predicting the state of the battery on the BMS to take safety precautionary measures. These solutions include both diagnostic and prognostic approaches e.g. with Kalman filter, particle filter, neural networks, parametric modeling, mapping, etc [231]. Through the use of the diagnostic and prognostic approaches on the BMS, it may be possible to derive a cost-effective and secure authentication solution for the physical security of the battery. Specifically it may be possible to derive a unique signature from the battery's physical features to authenticate the battery to detect/prevent counterfeit, swapping, and tampering. There are several sources which suggest that each battery has unique features [27, 237], but there are no related works on using them for the purpose of authentication.

Battery cells implemented in devices within IoT devices of the ITS ecosystem and other CPS are manufactured differently according to the application and design constraints. Moreover, their physical and chemical characteristics are unique in part because of the imperfections in the manufacturing process, where no two batteries are the exact same [237]. In fact, the battery features (e.g., internal resistance) have high variation from cell to cell under the same manufacturer and different manufacturers by 10-15% [27, 237]. Since counterfeit battery cells are manufactured differently from the original ones, their behavior may become significantly distinguishable in terms of voltage-current, thermal, and aging characteristics. Hence, the unique behavior of each battery cell may help in fingerprinting and authenticating the original battery cells and eliminating the counterfeit ones.

5.4.2 Related Work

Battery Management Systems (BMS) are implemented to evaluate the current state of the battery at run time using the generic manufacturer-provided models and measurements of the battery's voltage, current, temperature, and State-of-Charge (SoC). However, BMS face the challenge of precise computation of a state of the battery because of the generic battery models and therefore must have high thresholds for detecting unsafe states (such as overheating or overcharging). If the battery's precise characteristics are known, the online estimation may improve the functionality and safety of the ITS.

Moreover, existing battery authentication schemes are based on form factors, fixed ID, and challenge-response (highest cost), e.g., CRC or SHA1-HMAC [273]. The state-of-the-art schemes can be compromised via different techniques such as spoofing the form factors, RFIDs, or random number generator challenge-response. Furthermore, to implement such authentication schemes, additional hardware may be necessary to be embedded onto the battery pack (e.g., microcontrollers). For instance, the hardware chip and the integrated circuit that run the SHA-1 authentication scheme for battery cells can cost upto $2 [186, 66]. Hence, these solutions utilizing encryption or random number generation become less acceptable than the more traditional methods such as form factors or barcodes.

Exploiting physical variation to improve security has been leveraged in many security schemes in order to generate secret keys or unique identities [252, 227]. Moreover, the random physical variation of the wireless channel has been exploited to generate cryptographic symmetric keys for indoor, outdoor, and automotive wireless networks [184, 297, 285]. Furthermore, using biological features (e.g., iris, joints of a hand, thumbprint, and DNA) for authentication has been proposed [26]. Recently, high entropy signatures from the intrinsic impedance variation within the PCB has been used for its authentication [301].

In order to provide a novel battery security and authentication scheme solution with low overhead and high entropy fingerprinting, the exploitation of the measurable physical features of battery cells is proposed. The solution is applicable not just to IoT devices within ITS but also to other CPS.

Problem and Research Challenges

In summary, the design of battery security and authentication schemes for safety of ITS devices poses the following major challenges:

1. Estimation accuracy of the behavioral battery model used for the BMS and security schemes.
2. State-of-the-art schemes are at risk of being easily compromised due to their general security features.
3. Cost and complexity of implementing the schemes on the embedded system and circuit of the battery cells.

Novel Contributions and Overview of Concepts

The above-mentioned challenges are addressed with the following components of the physics-based battery security and authentication scheme:

1. **Physics-Based Fingerprinting (Section 5.4.3):** in which the behavior of a battery cell during charge/discharge cycles (Section 5.4.3) is monitored and its features (Section 5.4.3) are extracted as the battery fingerprint (see Figure 3.2).
2. **Authentication Scheme (Section 5.4.4):** that measures the features of the current battery and authenticates whether it is the original battery or it has been replaced/tampered (counterfeit). An experimental test-bed on real battery cells from Panasonic/LG is presented and a demonstration of the discharge temperature as the source of the fingerprinting is provided, indicating the potential applicability of this solution (Section 5.4.5).

5.4.3 Battery Fingerprinting

The following proposed methods assume the attack model in Section 5.2.5 (see Figure 5.4). In order to eliminate battery attacks and resolve the challenges of counterfeit, both quality testing and authentication methods have been developed. Main concerns for authentication by battery manufacturers include: 1) the size of the required embedded system; 2) the complexity of the algorithm and the required memory/computation power; and 3) cost of implementing the design.

Fingerprinting will be the responsibility of the battery manufacturer. Depending on the power consumption of the application, the battery may come in the form of a cell, module, or pack. However, the fingerprinting scheme is orthogonal to any type of battery. For fingerprinting, a data-driven (sensor measurement-based) modeling technique which requires the manufacturer to perform several standard tests on the battery. The resulting models are stored in the embedded system accessible by the CPS. These models will be integrated into the BMS in order to be used along with measurements to authenticate the battery.

Charge/Discharge Cycles

For a given battery (or battery pack), multiple charge and discharge cycles are conducted by the battery manufacturer. Multiple battery operating variables during the cycles are monitored to model the behavior of the battery and extract the features.

The standard charging scheme of a battery is shown in Figure 5.9. The battery is first charged using constant current (CC) until the maximum voltage of the battery. Then, it is charged using the constant voltage (CV) maintaining the maximum voltage of the battery until it fully charges.

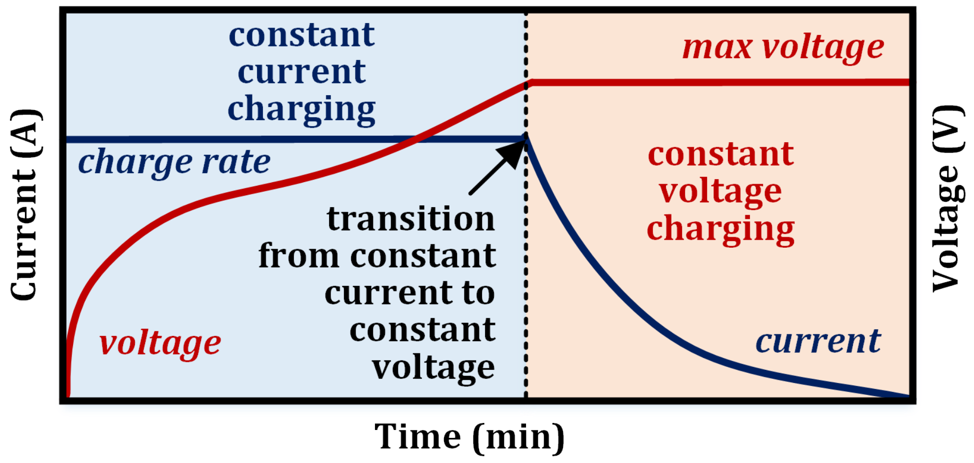

constant current charging
max voltage
charge rate
constant voltage charging
transition from constant current to constant voltage
voltage
current
Current (A)
Voltage (V)
Time (min)

Figure 5.9: CC-CV charging scheme of a battery.

The battery is charged using the constant current of 0.5C up to the maximum voltage. Afterwards, the battery is immediately discharged with the same rate of 0.5C to the minimum voltage. To reduce the amount of total time to extract the features of the battery, the constant voltage charging phase is skipped which can take approximately one additional hour. The C rate is the discharge rate in which the battery depletes in one hour. The initial temperature is the fixed room temperature (22°C). For instance, using four NCR18650B battery cells connected in series, the charge/discharge rate, maximum voltage, and minimum voltage are 1.67A, 16.8V, and 12V, respectively. By charging the battery up to the maximum voltage using the constant current, almost 80% of the battery capacity is charged according the battery datasheet. During the charge/discharge cycles, the voltage, current, and temperature values are measured and saved. Since there might be noise in the measured sensor values, moving average filter is applied. The voltage and current data will be used for evaluating the internal resistance of the battery.

The battery temperature affects the battery capacity and the voltage-current relationship. Since an active battery thermal management is mostly not available (especially at run time), the battery temperature cannot be maintained properly. However, due to the battery's internal resistance and capacity, the uncontrollable temperature can help in creating a unique fingerprint for it (higher entropy). Moreover, in order to extract the thermal features and correct the influence of the battery temperature, the battery is charged and discharged for various temperatures. To find out a temperature correction model, the battery is charged until the its temperature reached certain value (e.g., 28°C, 29°C, or 30°C) around the maximum voltage. It needs to be noted that the maximum voltage limitation is enforced to avoid over-charging. After reaching the required temperature, the battery is immediately discharged to the minimum voltage.

Feature Extraction

The unique behavior of the battery needs to be evaluated and modeled for the fingerprinting. Four time-series arrays are generated during each charge/discharge cycle: 1) charging voltage (cv), 2) charging temperature (ct), 3) discharging voltage (dv), and 4) discharging temperature (dt).

These four time-series arrays have different lengths due to the different initial capacities of the battery. Hence, in the pre-processing step, the smallest length of the time-series arrays is evaluated. All the arrays are cut and shifted to match the length for fair feature extraction and comparison in the later steps. The shifting of the time-series arrays improves the accuracy of the algorithm.

After gathering and pre-processing the data, the fingerprinting step is performed. In this step, there are two parts of classification and feature extraction. A given time-series of data (cycle) is classified based on its four features: 1) minimum, 2) maximum, 3) average, and 4) standard deviation. The cycles of the same battery which have almost the same features are classified into the same class. Hence, there will be a set of classes C of size $|C|$, where $C_i^v : v \in \{cv, dv, ct, dt\}$ corresponds to a class containing all the cycles having almost the same features (within a threshold defined by the standard deviation feature).

For each class C_i^v for variable v, a model $M(C_i^v)$ is created by extracting two types of feature vectors: magnitude ($|x|$) and rate of change (dx/dt). In the first feature vector, the mean magnitude is taken across all curves, stored in matrix X, that fall under $C_{v,i}$: ($AvgMagnitude = \overline{|X|}$). Then, in a similar way, the averaged rate of change value is computed across all curves, over groups of size, $Gsize$: ($AvgRoC = \overline{(X(t+Gsize) - X(t))/Gsize}$). Additionally, standard deviations are taken over the values to compute lower and upper threshold vectors, Thr_{lo} and Thr_{hi}, where $Thr_{lo}(M(C_{v,i})) = M(C_{v,i}$ - α*$std(M(C_{v,i}))$ and $Thr_{hi}(M(C_{v,i})) = M(C_{v,i}$ - α*$std(M(C_{v,i}))$.

Each rate of change value serves as a unique feature for the C_i and will be used for authenticating a battery pack (explained further in following section). Additionally, the standard deviations will be used to compute upper and lower thresholds (Thr_{lower} and Thr_{upper}) for the authentication step.

5.4.4 Proposed Authentication Scheme

Given any battery pack, the proposed authentication scheme will first follow similar steps as in the fingerprinting section. In either the designer or user side, the battery pack will be charged and then discharged with constant .2C rate and have its voltage and temperature measured at regular intervals. The resulting charging/discharging voltage and temperature curves for the current cycle, j, are labeled as cv_j, dv_j, ct_j, dt_j, respectively, and are shifted and filtered (median filter for temperature, and average filter for voltage) for fairer comparison and reduction of noise. According to the maximum, average and standard deviation values of each voltage and temperature curve, the curves are classified based on classes in C. If there is a curve that does not match any class, then that curve is considered as invalid and unusable for authentication.

Having classified each curve of cycle, j, of each variable, v, in $C_{v,i}$, feature extraction is then performed over the curves. As mentioned in the previous step, there are two primary features that are extracted along the curve: rate of change ($AvgRoC = (v_j(t+Gsize)-v_j(t))/Gsize$) and magnitude ($AvgMagnitude = |v_j(t)|$). An authentication step is performed for each profile by comparing these features with those in the model, $M(C_{v,i})$, corresponding to the class $C_{v,i}$ and corresponding thresholds, $Thr_{upper}(C_{v,i})$ and $Thr_{lower}(C_{v,i})$. If each feature is between the thresholds, then it is considered as an authenticated feature.

After comparison of all features of a single curve, total authenticity may be evaluated as *authenticity*, by dividing the number of authenticated features ($auth\_feat$) by the total number of features ($total\_feat$). If the overall *authenticity* is above a specific threshold, Thr_{auth}, then the tested battery pack is considered as an authentic one, otherwise it is not. To improve the accuracy, the authentication mechanism can use several cycles of data to calculate the overall *authenticity*. This type of scheme is mostly suitable for the designer stage (where there are more resources to perform the measurements) rather than the more constricted runtime stage (where battery authentication should not require more than one cycle).

Algorithm 1: Physics-Based Battery Fingerprinting Algorithm

```
   Input: Authentic Battery Measurement Vectors: $v$
   Input: Group Size: $Gsize$
   Input: Threshold Parameter: $\alpha$
   Input: Classification Parameter: $\beta$
   Output: Classes: $C$
   Output: Generated Fingerprint Models: $M$
   Output: Upper Thresholds: $Thr_{Hi}$
   Output: Lower Thresholds: $Thr_{Lo}$
 1 foreach $0 <= i < Length(v)$ do
 2     $v_i = Filter(v_i)$
 3     tempC = $med(v_i), min(v_i), max(v_i), avg(v_i), std(v_i)$
 4     // Add index of this measurement curve to closest matching class based on $\beta$
 5     $AddClass(C, tempC, i, \beta)$
 6 foreach $i < Length(C)$ do
 7     // Compute magnitude or rate of change features from all curves corresponding
        to each class
 8     $ind = C_i$
 9     $AvgMagnitudes = mean(v_{ind})$
10     $AvgRoC = mean(v_{ind}(t + Gsize) - v_{ind}(t))$
11     // Compute upper and lower thresholds for each feature based on variance
12     $Thr_{Hi}$,$Thr_{Lo}$ = $computeThr(v_{ind})$
13     $M_i = [AvgMagnitudes, AvgRoC, Thr_{Hi}, Thr_{Lo}]$
14 return $M$;
```

5.4.5 Battery Fingerprinting Experimentation

Experimental Setup

In order to conduct experiments and analyze the battery security and authentication scheme, a real battery test bed is set up as shown in Figure 5.10. The battery test bed setup comprises of: 1) a variable DC load (BK Precision 8514); 2) a variable DC power supply (Instek PSB-2400L2); 3) constant DC power supply; 4) multiple temperature sensors (LM35-DZ); and 5) up to four battery cells (Panasonic NCR18650B, LG HG2, SunLabz 18650B) (used in EVs such as those produced by Tesla). The requests for transmission of power (current) to the battery cells and the reception of sensor values (e.g., voltage, current, and temperature) are handled by Python functions. The functions utilize a serial communication between the computer and the devices in order to send commands and read values.

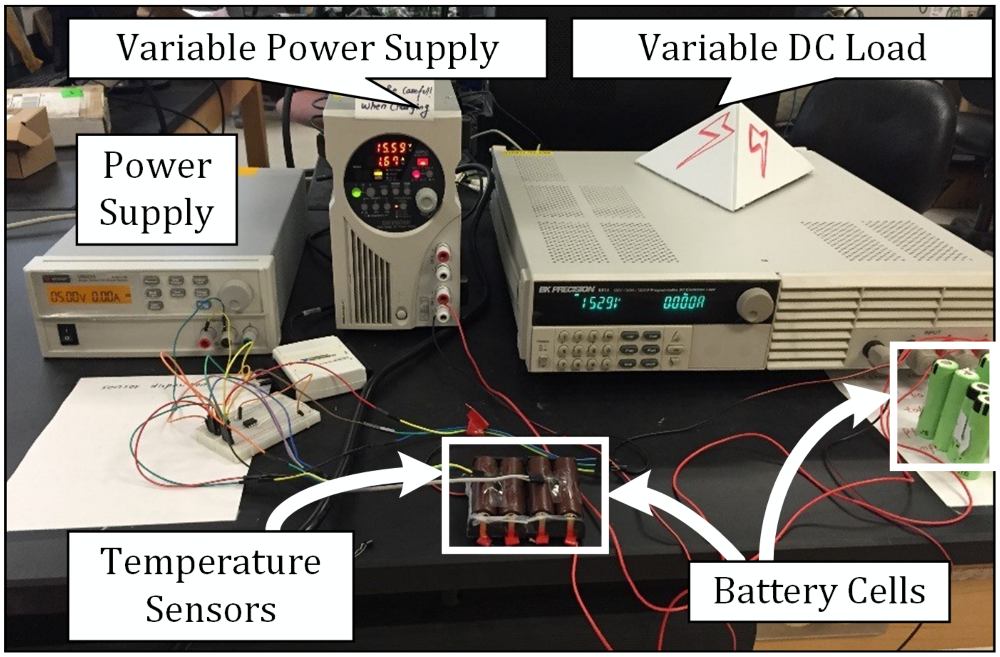


Figure 5.10: Experimental battery test bed.

The charging and discharging cycles (see Section 5.4.3) are managed by controllers implemented in MATLAB [2] through Python functions. Furthermore, the techniques and algorithms explained in Section 5.4.3 and 5.4.4 are implemented in MATLAB and leverage the saved data.

Proof of Concept

For a simple proof of concept, charge/discharge cycles were performed for seven different individual Panasonic batteries using the aforementioned test bed (instead of packs of four batteries). Features are extracted and converted from these cycles into threshold curves from the gathered sensor data. Out of the various threshold curves, the Discharge Temperature (*dt*) curves stood out the most. Figure 5.11 demonstrates the *dt*-based threshold curves (Class Upper and Lower Thresholds), as well as the extracted feature curves when the battery is authentic (a Panasonic battery) or fake (an LG battery). Notice how the Authentic Battery curve fits between the two threshold curves nicely, but the Fake Battery curve begins to break out from the region after the 700th sample or so. It was found that features extracted from the other sensor values did not appear to show as much promise regarding uniqueness. Hence, there is substantial room for research on discovering which physical characteristics (such as internal resistance) to use for fingerprinting and which features to extract from them.

It is important to note that both the fingerprinting method and authentication scheme are incomplete and they serve as a conceptual basis for other improved and advanced authentication schemes. However, this proof of concept demonstrates some promise that it might be feasible. Research challenges for improving the proposed solution include proper technology and sophisticated methods to gather information on the battery's physical characteristics, and ensuring that the fingerprinting and authentication mechanisms are tolerant to variation due to environmental factors and aging. More advanced methods such as particle filters and other offline/online feature extraction and classification machine learning techniques may assist in creating a solution for these challenges.

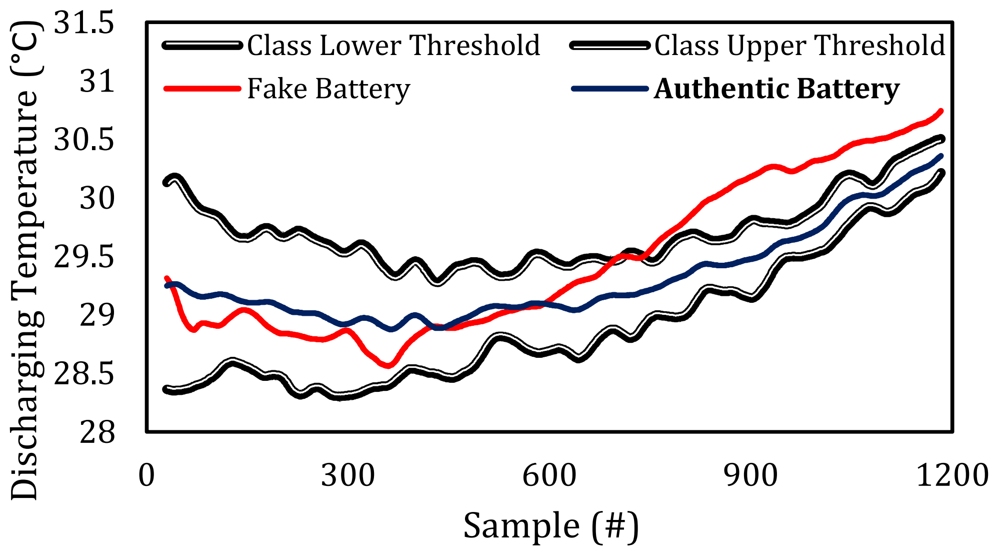


Figure 5.11: Proof of concept demonstrating that the extracted Discharge Temperature features from the Authentic (Panasonic NCR18650B) battery fits within the discharging temperature behavior fingerprint thresholds (based on Panasonic), whereas the ones from the Fake (LG H2) battery do not. Horizontal axis corresponds to discharge temperature sensor data samples gathered over the discharging time period.

5.5 Summary

To summarize, a comprehensive review and cross-layer security analysis of battery systems in the Internet of Things is presented with respect to ITS. Existing solutions tend to be ad-hoc and restricted to specific attack scenarios. A comprehensive security framework for battery systems in IoT devices within the ITS ecosystem comprising of cross-layer security analysis capability is required to prevent rising threats. In order to develop a holistic method to thwart battery attacks and prevent battery counterfeiting, the hardware, software and firmware should work independently and in conjunction with each other.

Finally, a novel physics-based battery security scheme is proposed to improve the security and safety of the IoT devices against battery attacks. In this scheme, the manufacturer generates a unique fingerprint of the battery out of extracted features from gathered sensor data during charging/discharging. Then, to assure that the battery has not been tampered with or swapped, another entity may authenticate the battery using the aforementioned fingerprint. This is one example and serves as a potential direction on how battery system security with respect to IoT may be improved.

Chapter 6

Conclusion

This dissertation has presented and addressed the following research challenges in the domain of Intelligent Transportation System (ITS) security: the need for a more thorough coverage of security challenges and existing solutions, the need for useful methodologies and tools for modeling and analyzing the impact of attacks and defenses, and the need for more effective security solutions to help prevent known and unforeseen attacks. The methodologies and tools in this dissertation focus on known ITS use cases, but they may serve as a foundation on how to study ITS security and deal with its challenges as both complexity and number of ITS use cases grow.

Other researchers, designers, and engineers may follow in similar steps provided in this dissertation to identify and define novel ITS security challenges, come up with ITS security modeling/analysis methods and tools for all related parties to benefit from, and design novel solutions. Such research will strengthen the ITS research community because it will be in more tune with the ongoing and unforeseen difficulties that ITS industry and national governments face. Following this, unique and powerful ITS solutions and architectures for both security and non-security purposes may be developed.

Bibliography

[1] Applied Reverse Engineering with IDA Pro. `https://resources.infosecinstitute.com/applied-reverse-engineering-ida-pro`.

[2] Matlab. `https://www.mathworks.com/products/matlab/`.

[3] MQTT. `http://mqtt.org`.

[4] IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 6: Wireless Access in Vehicular Environments. *IEEE Std 802.11p-2010 (Amendment to IEEE Std 802.11-2007 as amended by IEEE Std 802.11k-2008, IEEE Std 802.11r-2008, IEEE Std 802.11y-2008, IEEE Std 802.11n-2009, and IEEE Std 802.11w-2009)*, pages 1–51, July 2010.

[5] Car hacking research: Remote attack tesla motors. `https://keenlab.tencent.com/en/2016/09/19/Keen-Security-Lab-of-Tencent-Car-Hacking-Research-Remote-Attack-to-Tesla-Cars`, 2016.

[6] New car hacking research: 2017, remote attack tesla motors again. `https://keenlab.tencent.com/en/2017/07/27/New-Car-Hacking-Research-2017-Remote-Attack-Tesla-Motors-Again`, 2017.

[7] New vehicle security research by keenlab: Experimental security assessment of bmw cars. `https://keenlab.tencent.com/en/2018/05/22/New-CarHacking-Research-by-KeenLab-Experimental-Security-Assessment-of-BMW-Cars`, 2018.

[8] Experimental security research of tesla autopilot. `https://keenlab.tencent.com/en/whitepapers/Experimental_Security_Research_of_Tesla_Autopilot.pdf`, 2019.

[9] Keen security lab blog. `https://keenlab.tencent.com/en/`, 2019.

[10] M. Abrams and J. Weiss. Malicious control system cyber security attack case study–maroochy water services, australia. *McLean, VA: The MITRE Corporation*, 2008.

[11] I. Agadakos, C.-Y. Chen, M. Campanelli, et al. Jumping the air gap: Modeling cyber-physical attack paths in the internet-of-things. In *Proceedings of the 2017 Workshop on Cyber-Physical Systems Security and Privacy*, CPS '17, pages 37–48, New York, NY, USA, 2017. ACM.

[12] I. Ahmad, T. Kumar, M. Liyanage, J. Okwuibe, M. Ylianttila, and A. Gurtov. 5g security: Analysis of threats and solutions. In *2017 IEEE Conference on Standards for Communications and Networking (CSCN)*, pages 193–199. IEEE, 2017.

[13] N. Akhtar and A. Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access*, 6:14410–14430, 2018.

[14] G. A. Akpakwu, B. J. Silva, G. P. Hancke, and A. M. Abu-Mahfouz. A survey on 5g networks for the internet of things: Communication technologies and challenges. *IEEE Access*, 6:3619–3647, 2017.

[15] M. A. Al Faruque and K. Vatanparvar. Modeling, analysis, and optimization of Electric Vehicle HVAC systems. *Asia and South Pacific Design Automation Conference (ASP-DAC)*, pages 1–6, 2016.

[16] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE communications surveys & tutorials*, 17(4):2347–2376, 2015.

[17] R. Al-Mutiri, M. Al-Rodhaan, and Y. Tian. Improving vehicular authentication in vanet using cryptography. *International Journal of Communication Networks and Information Security*, 10(1):248–255, 2018.

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

[19] D. Andrea. *Battery Management Systems for Large Lithium-ion Battery Packs*. Artech house, 2010.

[20] S. Anthony. Tesla model s battery bursts into flames, car totally destroyed in 5 minutes. `http://arstechnica.com/cars/2016/08/tesla-model-s-france-battery-fire/`, 2016.

[21] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, et al. Understanding the mirai botnet. In *26th {USENIX} Security Symposium ({USENIX} Security 17)*, pages 1093–1110, 2017.

[22] S. D. Applegate. The dawn of kinetic cyber. In *2013 5th International Conference on Cyber Conflict (CYCON 2013)*, pages 1–15, June 2013.

[23] I. S. Association et al. Ieee 802.11 p: Ieee standard for information technology–local and metropolitan area networks–specific requirements–part 11: Wireless lan medium access control (mac) and physical layer (phy), 2014.

[24] J. Balasch, A. Rial, C. Troncoso, B. Preneel, I. Verbauwhede, and C. Geuens. Pretp: Privacy-preserving electronic toll pricing. In *USENIX Security Symposium*, volume 10, pages 63–78, 2010.

[25] A. Banerjee, K. K. Venkatasubramanian, T. Mukherjee, and S. K. S. Gupta. Ensuring safety, security, and sustainability of mission-critical cyber physical systems. *Proceedings of the IEEE*, 100(1):283–299, Jan 2012.

[26] A. Bansal, R. Agarwal, and R. K. Sharma. Statistical feature extraction based iris recognition system. *Sādhanā*, 41(5):507–518, 2016.

[27] Y. Barsukov. Battery selection, safety, and monitoring in mobile applications. 2005.

[28] A. Basak, S. Bhunia, and S. Ray. A Flexible Architecture for Systematic Implementation of SoC Security Policies. In *ICCAD*, pages 536–543, 2015.

[29] A. Basak, S. Bhunia, and S. Ray. Exploiting design-for-debug for flexible SoC security architecture. In *DAC*, pages 167:1–167:6, 2016.

[30] Y. O. Basciftci, F. Chen, J. Weston, R. Burton, and C. E. Koksal. How vulnerable is vehicular communication to physical layer jamming attacks? In *2015 IEEE 82nd Vehicular Technology Conference (VTC2015-Fall)*, pages 1–5. IEEE, 2015.

[31] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz. Sumo–simulation of urban mobility: an overview. In *Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation*. ThinkMind, 2011.

[32] A. Belmonte Martin, L. Marinos, E. Rekleitis, G. Spanoudakis, and N. Petroulakis. Threat landscape and good practice guide for software defined networks/5g. 2015.

[33] S. Bhunia, S. Ray, and S. Sur-Kolay. *Fundamentals of IP and SoC Security: Design, Validation, and Debug*. Springer, 2017.

[34] S. Bhunia and M. Tehranipoor. *The Hardware Trojan War: Attacks, Myths, and Defenses*. Springer, 2017.

[35] N. Bhushan, J. Li, D. Malladi, R. Gilmore, D. Brenner, A. Damnjanovic, R. T. Sukhavasi, C. Patel, and S. Geirhofer. Network densification: the dominant theme for wireless evolution into 5g. *IEEE Communications Magazine*, 52(2):82–89, 2014.

[36] V. Bibhu, R. Kumar, B. S. Kumar, and D. K. Singh. Performance analysis of black hole attack in vanet. *International Journal Of Computer Network and Information Security*, 4(11):47, 2012.

[37] I. Bilogrevic, I. Aad, P. Ginzboorg, V. Niemi, J.-P. Hubaux, et al. Track me if you can: On the effectiveness of context-based identifier changes in deployed mobile networks. In *Proceedings of the 19th Annual Network & Distributed System Security Symposium (NDSS 2012)*, number CONF. Internet Society, 2012.

[38] N. Bißmeyer, J. Njeukam, J. Petit, and K. M. Bayarou. Central misbehavior evaluation for vanets based on mobility data plausibility. In *Proceedings of the ninth ACM international workshop on Vehicular inter-networking, systems, and applications*, pages 73–82. ACM, 2012.

[39] S. Biswas, J. Mi, and V. Mi. Ddos attack on wave-enabled vanet through synchronization. In *2012 IEEE Global Communications Conference (GLOBECOM)*, pages 1079–1084. IEEE, 2012.

[40] B. Bloessl, M. Segata, C. Sommer, and F. Dressler. Towards an open source ieee 802.11 p stack: A full sdr-based transceiver in gnu radio. In *2013 IEEE Vehicular Networking Conference*, pages 143–149. IEEE, 2013.

[41] J. J. Blum and P. O. Okusun. Privacy implications of the traffic probe message service. In *13th International IEEE Conference on Intelligent Transportation Systems*, pages 342–347. IEEE, 2010.

[42] S. Bono, M. Green, A. Stubblefield, A. Juels, A. D. Rubin, and M. Szydlo. Security analysis of a cryptographically-enabled rfid device. In *USENIX Security Symposium*, volume 31, pages 1–16, 2005.

[43] M. Brandl, H. Gall, M. Wenger, V. Lorentz, M. Giegerich, F. Baronti, and G. F. et al. Batteries and battery management systems for electric vehicles. In *Design, Automation & Test in Europe Conference and Exhibition (DATE)*, pages 971–976. IEEE, 2012.

[44] C Miller, C Valasek. Securing Self-Driving Cars (one company at a time) , 2018. [Black Hat 2018, Briefing on Applied Self Driving Car Security].

[45] J. A. L. Calvo and R. Mathar. Secure blockchain-based communication scheme for connected vehicles. In *2018 European Conference on Networks and Communications (EuCNC)*, pages 347–351. IEEE, 2018.

[46] C. Cerrudo. Hacking us (and uk, australia, france, etc.) traffic control systems, 2013.

[47] C. Cerrudo. An emerging us (and world) threat: Cities wide open to cyber attacks. *Securing Smart Cities*, 2015.

[48] C. Cerrudo and D. Spaniel. Keeping smart cities smart: preempting emerging cyber attacks in us cities. *Institute for Critical Infrastructure Technology*, 2015.

[49] A. C.-F. Chan and J. Zhou. On smart grid cybersecurity standardization: Issues of designing with nistir 7628. *IEEE Communications Magazine*, 51(1):58–65, 2013.

[50] A. C.-F. Chan and J. Zhou. Cyber–physical device authentication for the smart grid electric vehicle ecosystem. *IEEE Journal on Selected Areas in Communications*, 32(7):1509–1517, 2014.

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

[52] S. Checkoway, D. McCoy, B. Kantor, et al. Comprehensive experimental analyses of automotive attack surfaces. *USENIX Security Symposium*, 2011.

[53] S. Checkoway, D. McCoy, B. Kantor, et al. Comprehensive experimental analyses of automotive attack surfaces. *USENIX Security Symposium*, 2011.

[54] C. Chen, X. Wang, W. Han, and B. Zang. A robust detection of the sybil attack in urban vanets. In *2009 29th IEEE International Conference on Distributed Computing Systems Workshops*, pages 270–276. IEEE, 2009.

[55] Q. A. Chen, Y. Yin, Y. Feng, et al. Exposing congestion attack on emerging connected vehicle based traffic signal control. In *Network and Distributed Systems Security (NDSS) Symposium 2018*, 2018.

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

[57] W. Chen, S. Ray, J. Bhadra, M. S. Abadir, and L. Wang. Challenges and Trends in Modern SoC Design Verification. *IEEE Design & Test*, 34(5):7–22, 2017.

[58] S. R. Chhetri and M. A. Al Faruque. Side channels of cyber-physical systems: Case study in additive manufacturing. *IEEE Design & Test*, 34(4):18–25, 2017.

[59] S. R. Chhetri, A. Canedo, and M. A. A. Faruque. Confidentiality breach through acoustic side-channel in cyber-physical additive manufacturing systems. *ACM Transactions on Cyber-Physical Systems*, 2(1):1–25, 2017.

[60] S. R. Chhetri, S. Faezi, and M. A. Al Faruque. Fix the leak! an information leakage aware secured cyber-physical manufacturing system. In *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*, pages 1408–1413. IEEE, 2017.

[61] S. R. Chhetri, J. Wan, and M. A. Al Faruque. Cross-domain security of cyber-physical systems. In *2017 22nd Asia and South Pacific Design Automation Conference (ASP-DAC)*, pages 200–205. IEEE, 2017.

[62] K.-T. Cho and K. G. Shin. Fingerprinting electronic control units for vehicle intrusion detection. In *USENIX Security Symposium*, 2016.

[63] J. Choi and S. Jung. A security framework with strong non-repudiation and privacy in vanets. In *2009 6th IEEE Consumer Communications and Networking Conference*, pages 1–5. IEEE, 2009.

[64] G. Clark, M. Doran, and W. Glisson. A malicious attack on the machine learning policy of a robotic system. In *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pages 516–521. IEEE, 2018.

[65] CNET. Here's why samsung note 7 phones are catching fire - CNET. `https://www.cnet.com/news/why-is-samsung-galaxy-note-7-exploding-overheating/`, 2016.

[66] M. Conner. Friend or foe: Battery-authentication ics separate the good guys from the bad. `http://tayloredge.com/reference/Batteries/EDN-BatteryAuthentication.pdf`, 2006.

[67] M. Conner. Friend or foe: Battery-authentication ics separate the good guys from the bad-all battery packs are not created equal: Unauthorized after-market packs may contain cells that can self-destruct when. *EDN*, 51(3):59–64, 2006.

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

[69] S. Daneshmand, A. Jafarnia-Jahromi, A. Broumandan, and G. Lachapelle. A low-complexity gps anti-spoofing method using a multi-antenna array. *aa*, 2:2, 2012.

[70] D. Davidson, H. Wu, R. Jellinek, V. Singh, and T. Ristenpart. Controlling uavs with sensor input spoofing attacks. In *10th {USENIX} Workshop on Offensive Technologies ({WOOT} 16)*, 2016.

[71] S. L. Dennisen and J. P. Müller. Agent-based voting architecture for traffic applications. In *German Conference on Multiagent System Technologies*, pages 200–217. Springer, 2015.

[72] Overview of the doe advanced battery research and development program. `http://energy.gov/sites/prod/files/2014/09/f18/battery_rd_amr_plenary\_june_2014_final.pdf`, September 2014.

[73] T. Dobbyn. U.S. opening formal probe into GM Volt fire risk. `https://www.reuters.com/article/us-gm-volt/u-s-opening-formal-probe-into-gm-volt-fire-risk-idUSTRE7A01SH20111126`, Sep 2011.

[74] M. M. Dobersek. An operational comparison of pre-time, semi-actuated, and fully actuated interconnected traffic control signal systems. 1998.

[75] J. R. Douceur. The sybil attack. In *International workshop on peer-to-peer systems*, pages 251–260. Springer, 2002.

[76] D. Engels, X. Fan, G. Gong, H. Hu, and E. M. Smith. Hummingbird: ultra-lightweight cryptography for resource-constrained devices. In *International Conference on Financial Cryptography and Data Security*, pages 3–18. Springer, 2010.

[77] ETSI. Intelligent transport systems (its); vehicular communications; basic set of applications; part 2: Specification of cooperative awareness basic service etsi standard rev. v1.3.0 draft, 2013.

[78] I. ETSI. Intelligent transport systems (its); security; threat, vulnerability and risk analysis (tvra). Technical report, ETSI TR 102 893, European Telecommunications Standards Institute, 2010.

[79] I. Evtimov, K. Eykholt, E. Fernandes, et al. Robust physical-world attacks on machine learning models. *CoRR*, abs/1707.08945, 2017.

[80] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song. Robust Physical-World Attacks on Deep Learning Models. *arXiv*, Jul 2017.

[81] S. Faezi, S. R. Chhetri, A. V. Malawade, J. C. Chaput, W. Grover, P. Brisk, and M. A. Al Faruque. Oligo-snoop: a non-invasive side channel attack against dna synthesis machines. In *Network and Distributed Systems Security (NDSS) Symposium 2019*, 2019.

[82] M. Feiri, J. Petit, and F. Kargl. Efficient and secure storage of private keys for pseudonymous vehicular communication. In *Proceedings of the 2013 ACM workshop on Security, privacy & dependability for cyber vehicles*, pages 9–18. ACM, 2013.

[83] A. Filippi, K. Moerman, V. Martinez, A. Turley, O. Haran, and R. Toledano. Ieee 802.11 p ahead of lte-v2v for safety applications. *Autotalks NXP*, 2017.

[84] U. Fiore, F. Palmieri, A. Castiglione, V. Loia, and A. De Santis. Multimedia-based battery drain attacks for android devices. In *2014 IEEE 11th Consumer Communications and Networking Conference (CCNC)*, pages 145–150. IEEE, 2014.

[85] I. D. Foster, A. Prudhomme, K. Koscher, and S. Savage. Fast and vulnerable: A story of telematic failures. In *USENIX Workshop on Offensive Technologies*, 2015.

[86] N. S. Foundation. Computer and network systems (cns): Core programs program solicitation nsf 18-569, 2018.

[87] A. Francillon, B. Danev, and S. Capkun. Relay attacks on passive keyless entry and start systems in modern cars. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*. Eidgenössische Technische Hochschule Zürich, Department of Computer Science, 2011.

[88] Q. Gan. *Macroscopic modeling and analysis of urban vehicular traffic.* PhD thesis, UC Irvine, 2014.

[89] Q.-J. Gan, W.-L. Jin, and V. V. Gayah. Analysis of traffic statics and dynamics in signalized networks: a poincaré map approach. *Transportation Science*, 51(3):1009–1029, 2017.

[90] F. Gardner. Iran shows film of captured us drone. *BBC News*, 2011.

[91] M. T. Garip, M. E. Gursoy, P. Reiher, and M. Gerla. Congestion attacks to autonomous cars using vehicular botnets. In *NDSS Workshop on Security of Emerging Networking Technologies (SENT), San Diego, CA*, 2015.

[92] K. E. Gemeinschaften. *White paper-European transport policy for 2010: time to decide.* Office for Official Publications of the European Communities, 2001.

[93] S. Georgi. Counterfeit cell phone and laptop batteries: Caution, credibility, causes and cures. `http://www.tayloredge.com/reference/Batteries/CounterfeitBatteries.pdf`, 2004.

[94] M. Ghaderi, D. Goeckel, A. Orda, and M. Dehghan. Efficient wireless security through jamming, coding and routing. In *2013 IEEE International Conference on Sensing, Communications and Networking (SECON)*, pages 505–513. IEEE, 2013.

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

[96] B. Ghena, W. Beyer, A. Hillaker, et al. Green lights forever: Analyzing the security of traffic infrastructure. *WOOT*, 14:7–7, 2014.

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

[98] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of computer and system sciences*, 28(2):270–299, 1984.

[99] P. Golle, D. Greene, and J. Staddon. Detecting and correcting malicious data in vanets. In *Proceedings of the 1st ACM international workshop on Vehicular ad hoc networks*, pages 29–37. ACM, 2004.

[100] A. Greenberg. After jeep hack, chrysler recalls 1.4m vehicles for bug fix. 2015.

[101] S. J. Greenwald. Discussion Topic: What is the Old Security Paradigm. In *Workshop on New Security Paradigms*, pages 107–118, 1998.

[102] I. A. Grzemba. *MOST: the automotive multimedia network*. Franzis Verlag, 2012.

[103] Y. Gu, Z. Qian, and G. Zhang. Traffic state estimation for urban road networks using a link queue model. *Transportation Research Record*, 2623(1):29–39, 2017.

[104] G. Guette and C. Bryce. Using tpms to secure vehicular ad-hoc networks (vanets). In *IFIP International Workshop on Information Security Theory and Practices*, pages 106–116. Springer, 2008.

[105] U. Guin, D. DiMase, and M. Tehranipoor. Counterfeit integrated circuits: Detection, avoidance, and the challenges ahead. *Journal of Electronic Testing*, 30(1):9–23, 2014.

[106] U. Guin, D. DiMase, and M. Tehranipoor. Counterfeit integrated circuits: detection, avoidance, and the challenges ahead. *Journal of Electronic Testing*, 30(1):9–23, 2014.

[107] Y. Hao, J. Tang, and Y. Cheng. Cooperative sybil attack detection for position based applications in privacy preserved vanets. In *2011 IEEE Global Telecommunications Conference-GLOBECOM 2011*, pages 1–5. IEEE, 2011.

[108] S. K. Harit, G. Singh, and N. Tyagi. Fox-hole model for data-centric misbehaviour detection in vanets. In *2012 Third International Conference on Computer and Communication Technology*, pages 271–277. IEEE, 2012.

[109] M. Harris. Researcher hacks self-driving car sensors. *IEEE Spectrum*, 9, 2015.

[110] C. Harsch, A. Festag, and P. Papadimitratos. Secure position-based routing for vanets. In *2007 IEEE 66th Vehicular Technology Conference*, pages 26–30. IEEE, 2007.

[111] J. Harshan, S. Chang, and Y. Hu. Insider-attacks on physical-layer group secret-key generation in wireless networks. *CoRR*, abs/1701.03568, 2017.

[112] K. Hartmann and C. Steup. The vulnerability of uavs to cyber attacks - an approach to the risk assessment. In *2013 5th International Conference on Cyber Conflict (CYCON 2013)*, pages 1–23, June 2013.

[113] M. A. Hasbini, C. Cerrudo, D. Jordan, et al. The smart city department cyber security role and implications. *Securing Smart Cities*, 2016.

[114] H. Hasbullah, I. A. Soomro, et al. Denial of service (dos) attack and its possible solutions in vanet. *World Academy of Science, Engineering and Technology, International Journal of Electrical, Computer, Energetic, Electronic and Communication Engineering*, 4(5):813–817, 2010.

[115] M. Heller. Pegasus ios exploit uses three zero days to attack high-value targets. `https://searchsecurity.techtarget.com/news/450303267/Pegasus-iOS-exploit-uses-three-zero-days-to-attack-high-value-targets`. (Accessed on 09/19/2019).

[116] J. Hortelano, J. C. Ruiz, and P. Manzoni. Evaluating the usefulness of watchdogs for intrusion detection in vanets. In *2010 IEEE International Conference on Communications Workshops*, pages 1–5. IEEE, 2010.

[117] M. M. Hossain, M. Fotouhi, and R. Hasan. Towards an analysis of security issues, challenges, and open problems in the internet of things. In *Services (SERVICES), 2015 IEEE World Congress on*, pages 21–28. IEEE, 2015.

[118] P. Hosseini and K. Savla. A comparison study between proportionally fair and max pressure controllers for signalized arterial networks. Technical report, 2016.

[119] C. Hu, B. D. Youn, and J. Chung. A multiscale framework with extended kalman filter for lithium-ion battery {SOC} and capacity estimation. *Applied Energy*, 92:694 – 704, 2012.

[120] L. Huang and Q. Yang. Low-cost gps simulator gps spoofing by sdr, 2015.

[121] J. P. Hubaux, S. Capkun, and J. Luo. The security and privacy of smart vehicles. *IEEE Security Privacy*, 2(3):49–55, May 2004.

[122] T. Huddleston Jr. These chinese hackers tricked tesla's autopilot into suddenly switching lanes. `https://www.cnbc.com/2019/04/03/chinese-hackers-tricked-teslas-autopilot-into-switching-lanes.html`.

[123] T. E. Humphreys. Detection strategy for cryptographic gnss anti-spoofing. *IEEE Transactions on Aerospace and Electronic Systems*, 49(2):1073–1090, 2013.

[124] T. E. Humphreys, B. M. Ledvina, M. L. Psiaki, B. W. O'Hanlon, and P. M. Kintner. Assessing the spoofing threat: Development of a portable gps civilian spoofer. In *Radionavigation laboratory conference proceedings*, 2008.

[125] T. E. Humphreys, B. M. Ledvina, M. L. Psiaki, B. W. O'Hanlon, and P. M. Kintner. Assessing the spoofing threat: Development of a portable gps civilian spoofer. In *Radionavigation laboratory conference proceedings*, 2008.

[126] R. Hussain and S. Zeadally. Autonomous cars: Research results, issues and future challenges. *IEEE Communications Surveys & Tutorials*, 2018.

[127] O. A. Ibrahim, A. M. Hussain, G. Oligeri, and R. Di Pietro. Key is in the air: Hacking remote keyless entry systems. 2018.

[128] J. T. Isaac, J. S. Camara, S. Zeadally, and J. T. Marquez. A secure vehicle-to-roadside communication payment protocol in vehicular ad hoc networks. *Computer Communications*, 31(10):2478–2484, 2008.

[129] R. M. Ishtiaq Roufa, H. Mustafaa, S. O. Travis Taylora, W. Xua, M. Gruteserb, W. Trappeb, and I. Seskarb. Security and privacy vulnerabilities of in-car wireless networks: A tire pressure monitoring system case study. In *19th USENIX Security Symposium, Washington DC*, pages 11–13, 2010.

[130] R. M. Ishtiaq Roufa, H. Mustafaa, S. O. Travis Taylora, W. Xua, M. Gruteserb, W. Trappeb, and I. Seskarb. Security and privacy vulnerabilities of in-car wireless networks: A tire pressure monitoring system case study. *19th USENIX Security Symposium, Washington DC*, pages 11–13, 2010.

[131] G. A. Jacoby, R. Marchany, and N. Davis. Using battery constraints within mobile hosts to improve network security. *IEEE Security & Privacy*, 4(5):40–49, 2006.

[132] T. Jeske. 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, 2013.

[133] H.-Y. Jin and W.-L. Jin. Control of a lane-drop bottleneck through variable speed limits. *Transportation Research Part C: Emerging Technologies*, 58:568–584, 2015.

[134] W. Jin. Kinematic wave models of network vehicular traffic. *arXiv preprint math/0309060*, 2003.

[135] W.-L. Jin. A link queue model of network traffic flow. *arXiv preprint arXiv:1209.2361*, 2012.

[136] W.-L. Jin. Point queue models: A unified approach. *Transportation Research Part B: Methodological*, 77:1–16, 2015.

[137] W.-L. Jin and H. Jin. Analysis and design of a variable speed limit control system at a freeway lane-drop bottleneck: A switched systems approach. In *53rd IEEE Conference on Decision and Control*, pages 1753–1758. IEEE, 2014.

[138] W.-L. Jin and J. Laval. Bounded acceleration traffic flow models: A unified approach. *Transportation Research Part B: Methodological*, 111:1–18, 2018.

[139] W.-L. Jin and Y. Yu. Performance analysis and signal design for a stationary signalized ring road. *arXiv preprint arXiv:1510.01216*, 2015.

[140] A. Jindal, A. Pathak, Y. C. Hu, and S. Midkiff. On death, taxes, and sleep disorder bugs in smartphones. In *Proceedings of the Workshop on Power-Aware Computing and Systems*, page 1. ACM, 2013.

[141] M. Julia Carrillo. Robotic cars test platform for connected and automated vehicles. Master's thesis, 2015. Copyright - Database copyright ProQuest LLC; ProQuest does not claim copyright in the individual underlying works; Last updated - 2016-03-28.

[142] Kali Tools. Binwalk Penetration Testing Tools. `https://tools.kali.org/forensics/binwalk`.

[143] S. Kamkar. Drive it like you hacked it: New attacks and tools to wirelessly steal cars. *Presentation at DEFCON*, 23, 2015.

[144] M. Kane. Panasonic to supply tesla with 2 billion lithium-ion battery cells from 2014 to 2017. `http://insideevs.com/panasonic-to-supply-tesla-with\-2-billion-lithium-ion-battery-cells-from-2014-to-2017/`, 2013.

[145] G. Karimi and X. Li. Thermal management of lithium-ion batteries for electric vehicles. *Journal of Energy Research*, pages 13–24, 2013.

[146] S. Karnouskos. Stuxnet worm impact on industrial cyber-physical system security. In *IECON 2011-37th Annual Conference of the IEEE Industrial Electronics Society*, pages 4490–4494. IEEE, 2011.

[147] H. Kaur, S. Batish, and A. Kakaria. An approach to detect the wormhole attack in vehicular adhoc networks. *Int. J. Smart Sens. Ad Hoc Netw*, 4:86–89, 2012.

[148] K. B. Kelarestaghi, K. Heaslip, and R. Gerdes. Vehicle security: Risk assessment in transportation. *arXiv preprint arXiv:1804.07381*, 2018.

[149] L. Kelion. Nissan Leaf electric car hack revealed. `https://www.bbc.com/news/technology-35642749`, Sep 2019.

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

[151] A. J. Kerns, D. P. Shepard, J. A. Bhatti, and T. E. Humphreys. Unmanned aircraft capture and control via gps spoofing. *Journal of Field Robotics*, 31(4):617–636, 2014.

[152] R. Khatoun, P. Gut, R. Doulami, L. Khoukhi, and A. Serhrouchni. A reputation system for detection of black hole attack in vehicular networking. In *2015 International Conference on Cyber Security of Smart Cities, Industrial Control System and Communications (SSIC)*, pages 1–5. IEEE, 2015.

[153] G.-H. Kim, K.-H. Lee, S.-S. Kim, and J.-M. Kim. Vehicle relay attack avoidance methods using rf signal strength. *Communications and Network*, 5(03):573, 2013.

[154] N. Koblitz. The uneasy relationship between mathematics and cryptography. *Notices of the AMS*, 54(8):972–979, 2007.

[155] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas. Ddos in the iot: Mirai and other botnets. *Computer*, 50(7):80–84, 2017.

[156] Girl's iphone bursts into flames mid-flight. i thought we were going down. `http://komonews.com/news/local/girls-iphone-bursts-into-flames-mid-flight-i-thought-we-were-going-down`, 2011.

[157] P. Koonce, L. Rodegerdts, K. Lee, et al. Traffic signal timing manual. us department of transportation, federal highway administration, 1200 new jersey avenue, se washington, dc 20590, publication number fhwa-hop-08-024, june 2008, 2008.

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

[159] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial examples in the physical world. *arXiv*, Jul 2016.

[160] D. Kushwaha, P. K. Shukla, and R. Baraskar. A survey on sybil attack in vehicular ad-hoc network. *International Journal of Computer Applications*, 98(15), 2014.

[161] L Johnson, M Fitzsimmons. Uber self-driving cars: Everything you need to know. `https://www.techradar.com/news/uber-self-driving-cars`, 2018. [Online article on Techradar].

[162] F. Lambert. Tesla Model S caught fire and burned down while charging at a Supercharger. `https://electrek.co/2016/01/01/tesla-model-s-caught-fire-and-burned-down-charging-supercharger`, Jan 2016.

[163] F. Lambert. Tesla vehicle caught on fire while plugged in at Supercharger station - Electrek. `https://electrek.co/2019/06/01/tesla-fire-supercharger`, Jun 2019.

[164] A. Laszka, B. Potteiger, Y. Vorobeychik, et al. Vulnerability of transportation networks to traffic-signal tampering. In *Proceedings of the 7th International Conference on Cyber-Physical Systems*, page 16. IEEE Press, 2016.

[165] B. Lawson. Buying batteries in china. `http://www.mpoweruk.com/china_batteries.pdf`, 2017.

[166] N. Lawson. Highway to hell: Hacking toll systems. *Presentation at Blackhat*, 2008.

[167] R. Leale and K. Sireci Renner. Car hacking village: Securing critical automotive systems. `https://www.carhackingvillage.com/`, 2019.

[168] P. LeBeau. Waymo hits 10 millionth mile, prepares for public ride hailing. `https://www.cnbc.com/2018/10/10/waymo-hits-10-millionth-mile-prepares-for-public-ride-hailing.html`, 2018. [Online article on CNBC].

[169] K. C. Lee, S.-H. Lee, R. Cheung, U. Lee, and M. Gerla. First experience with cartorrent in a real vehicular ad hoc network testbed. In *2007 Mobile Networking for Vehicular Environments*, pages 109–114. IEEE, 2007.

[170] J. Leyden. Polish teen derails tram after hacking train network: Turns city network into hornby set. 2008.

[171] A. Lima, F. Rocha, M. Völp, and P. Esteves-Veríssimo. Towards safe and secure autonomous and cooperative vehicle ecosystems. In *Proceedings of the 2nd ACM Workshop on Cyber-Physical Systems Security and Privacy*, pages 59–70. ACM, 2016.

[172] X. Lin, P. Bogdan, N. Chang, and M. Pedram. Machine learning-based energy management in a hybrid electric vehicle to minimize total operating cost. *2015 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 627–634, Nov 2015.

[173] X. Lin, R. Lu, C. Zhang, H. Zhu, P.-H. Ho, and X. Shen. Security in vehicular ad hoc networks. *IEEE communications magazine*, 46(4):88–95, 2008.

[174] C. B. Liu, B. Sadeghi, and E. W. Knightly. Enabling vehicular visible light communication (v2lc) networks. In *Proceedings of the Eighth ACM international workshop on Vehicular inter-networking*, pages 41–50. ACM, 2011.

[175] M. Liyanage, I. Ahmad, A. B. Abro, A. Gurtov, and M. Ylianttila. *A Comprehensive Guide to 5G Security*. John Wiley & Sons, 2018.

[176] A. B. Lopez. *Physical layer key generation for wireless communication security in automotive cyber-physical systems*. PhD thesis, UC Irvine, 2017.

[177] A. B. Lopez, K. Vatanparvar, A. P. D. Nath, S. Yang, S. Bhunia, and M. A. Al Faruque. A security perspective on battery systems of the internet of things. *Journal of Hardware and Systems Security*, 1(2):188–199, 2017.

[178] L. Lu, X. Han, J. Li, J. Hua, and M. Ouyang. A review on the key issues for lithium-ion battery management in electric vehicles. *Journal of power sources*, 226:272–288, 2013.

[179] S. Lu, Y. Yao, and W. Shi. Collaborative learning on the edges: A case study on connected vehicles. In *2nd {USENIX} Workshop on Hot Topics in Edge Computing (HotEdge 19)*, 2019.

[180] S. Lucero. ihs-technology-whitepaper-cellular-vehicle-to-everything-c-v2x-connectivity.pdf. `https://www.qualcomm.com/media/documents/files/ihs-technology-whitepaper-cellular-vehicle-to-everything-c-v2x-connectivity.pdf`, June 2016. (Accessed on 09/19/2019).

[181] S. Lukic and Z. Pantic. Cutting the cord: Static and dynamic inductive wireless charging of electric vehicles. *IEEE Electrification Magazine*, 1(1):57–64, 2013.

[182] R. Makowitz and C. Temple. Flexray-a communication network for automotive control systems. In *2006 IEEE International Workshop on Factory Communication Systems*, pages 207–212. IEEE, 2006.

[183] T. R. Markham and A. Chernoguzov. A balanced approach for securing the obd-ii port. *SAE International Journal of Passenger Cars-Electronic and Electrical Systems*, 10(2017-01-1662):390–399, 2017.

[184] S. Mathur, W. Trappe, N. Mandayam, C. Ye, and A. Reznik. Radio-telepathy: extracting a secret key from an unauthenticated wireless channel. In *Proceedings of the 14th ACM international conference on Mobile computing and networking*, pages 128–139, 2008.

[185] S. Meiklejohn, K. Mowery, S. Checkoway, and H. Shacham. The phantom tollbooth: Privacy-preserving electronic toll collection in the presence of driver collusion. In *USENIX security symposium*, volume 201, 2011.

[186] R. Merritt. Fake batteries blow up in the industry's face. `http://www.eetimes.com/document.asp?doc_id=1151288`, September 2004.

[187] C. Miller. Battery firmware hacking: Inside the innards of a smart battery. *Black Hat USA*, 2011.

[188] C. Miller and C. Valasek. A survey of remote automotive attack surfaces. *Black Hat USA*, 2014:94, 2014.

[189] C. Miller and C. Valasek. Car hacking: the definitive source. `http://illmatics.com/content.zip`, 2015.

[190] C. Miller and C. Valasek. Remote exploitation of an unaltered passenger vehicle. *Black Hat USA*, 2015:91, 2015.

[191] A. Millner. Modeling Lithium Ion Battery degradation in electric vehicles. *IEEE Conference on Innovative Technologies for an Efficient and Reliable Electricity Supply*, pages 349–356, 2010.

[192] P. Mishra, S. Bhunia, and M. Tehranipoor. *Hardware IP Security and Trust*. Springer, 2017.

[193] R. N. Mitra and D. P. Agrawal. 5g mobile technology: A survey. *ICT Express*, 1(3):132–137, 2015.

[194] B. R. Moyers, J. P. Dunning, R. C. Marchany, and J. G. Tront. Effects of wi-fi and bluetooth battery exhaustion attacks on mobile devices. In *43rd Hawaii International Conference on System Sciences (HICSS)*, pages 1–9. IEEE, 2010.

[195] T. Moynihan. Why hoverboards keep exploding. `http://www.wired.com/2015/12/why-hoverboards-keep-exploding/`, December 2015.

[196] A. Narayanan, N. Thiagarajan, M. Lakhani, M. Hamburg, D. Boneh, et al. Location privacy via private proximity testing. In *NDSS*, volume 11, 2011.

[197] D. C. Nash, T. L. Martin, D. S. Ha, and M. S. Hsiao. Towards an intrusion detection system for battery exhaustion attacks on mobile computing devices. In *Pervasive Computing and Communications Workshops, 2005. PerCom 2005 Workshops. Third IEEE International Conference on*, pages 141–145. IEEE, 2005.

[198] Auxiliary power unit battery fire japan airlines boeing 787-8, ja829j. `http://www.ntsb.gov/investigations/accidentreports/pages/AIR1401.aspx`, 2013.

[199] C. W. O'donnell, G. E. Suh, and S. Devadas. Puf-based random number generation. *In MIT CSAIL CSG Technical Memo*, 2004.

[200] U. S. D. of Transportation. About its standards: Its standards and the u.s. dot its research initiatives. Technical report, 2018.

[201] U. D. of Transportation: Federal Highway Administration. Adaptive signal control technologies. Technical report, 2011.

[202] T. P. Oman and K. J. Hawes. Relay attack prevention for passive entry passive start (peps) vehicle security systems, Jan. 6 2015. US Patent 8,930,045.

[203] H. Onishi. Paradigm change of vehicle cyber security. In *2012 4th International Conference on Cyber Conflict (CYCON 2012)*, pages 1–11. IEEE, 2012.

[204] A. Osseiran, F. Boccardi, V. Braun, K. Kusume, P. Marsch, M. Maternia, O. Queseth, M. Schellmann, H. Schotten, H. Taoka, et al. Scenarios for 5g mobile and wireless communications: the vision of the metis project. *IEEE communications magazine*, 52(5):26–35, 2014.

[205] N. Owano. Rq-170 drone's ambush facts spilled by iranian engineer. *Phys. org*, 2011.

[206] M. Papageorgiou, C. Diakaki, V. Dinopoulou, et al. Review of road traffic control strategies. *Proceedings of the IEEE*, 91(12):2043–2067, 2003.

[207] S. Park, B. Aslam, D. Turgut, and C. C. Zou. Defense against sybil attack in vehicular ad hoc network based on roadside unit support. In *MILCOM 2009-2009 IEEE Military Communications Conference*, pages 1–7. IEEE, 2009.

[208] J. Petit and S. E. Shladover. Potential cyberattacks on automated vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 16(2):546–556, 2015.

[209] J. Petit, B. Stottelaar, M. Feiri, and F. Kargl. Remote attacks on automated vehicles sensors: Experiments on camera and lidar. *Black Hat Europe*, 11:2015, 2015.

[210] J. Petit, B. Stottelaar, M. Feiri, and F. Kargl. Remote attacks on automated vehicles sensors: Experiments on camera and lidar. *Black Hat Europe*, 11:2015, 2015.

[211] S. Plosz and P. Varga. Security and safety risk analysis of vision guided autonomous vehicles. In *2018 IEEE Industrial Cyber-Physical Systems (ICPS)*, pages 193–198. IEEE, 2018.

[212] Y. Qian, K. Lu, and N. Moayeri. A secure vanet mac protocol for dsrc applications. In *IEEE GLOBECOM 2008-2008 IEEE Global Telecommunications Conference*, pages 1–5. IEEE, 2008.

[213] B. Qin, Q. Wu, J. Domingo-Ferrer, and L. Zhang. Preserving security and privacy in large-scale vanets. In *International Conference on Information and Communications Security*, pages 121–135. Springer, 2011.

[214] A. Rawat, S. Sharma, and R. Sushil. Vanet: Security attacks and its possible solutions. *Journal of Information and Operations Management*, 3(1):301, 2012.

[215] S. Ray. Transportation security in the era of autonomous vehicles: Challenges and practice. In *ICCAD 2017*, 2017.

[216] S. Ray. Safety, Security, and Reliability: The Automotive Robustness Problem and an Architectural Solution. In *ICCE 2019*, 2019.

[217] S. Ray, W. Chen, J. Bhadra, and M. A. A. Faruque. Extensibility in Automotive Security: Current Practice and Challenges. In *DAC 2017*, 2017.

[218] S. Ray, W. Chen, and R. Cammarota. Protecting the Supply Chain of Automotives and IoTs. In *DAC 2018*, pages 89:1–89:4, 2017.

[219] S. Ray, E. Peeters, M. Tehranipoor, and S. Bhunia. System-on-Chip Platform Security Assurance: Architecture and Validation. *Proceedings of the IEEE*, 106(1):21–37, 2018.

[220] J. Reilly, S. Martin, M. Payer, and A. Bayen. On cybersecurity of freeway control systems: Analysis of coordinated ramp metering attacks. *Transportation Research, Part B*, 2014.

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

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

[223] J. Reilly, S. Martin, M. Payer, et al. On cybersecurity of freeway control systems: Analysis of coordinated ramp metering attacks 2. In *Transportation Research Board 94th Annual Meeting*, 1755.

[224] R. P. Roess, E. S. Prassas, and W. R. McShane. *Traffic engineering, Fifth Edition.* Pearson/Prentice Hall, 2019.

[225] P. Rogaway. Practice-oriented provable security and the social construction of cryptography. *Unpublished essay*, 2009.

[226] C. Rohner, S. Raza, D. Puccinelli, and T. Voigt. Security in visible light communication: Novel challenges and opportunities. *Sensors & Transducers Journal*, 192(9):9–15, 2015.

[227] M. Rostami, J. B. Wendt, M. Potkonjak, and F. Koushanfar. Quo vadis, puf?: trends and challenges of emerging physical-disorder based security. *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition 2014 (DATE'14)*, page 352, 2014.

[228] M. Ruff. Evolution of local interconnect network (lin) solutions. In *2003 IEEE 58th Vehicular Technology Conference. VTC 2003-Fall (IEEE Cat. No. 03CH37484)*, volume 5, pages 3382–3389. IEEE, 2003.

[229] F. Sagstetter, M. Lukasiewycz, S. Steinhorst, M. Wolf, A. Bouard, W. R. Harris, S. Jha, T. Peyrin, A. Poschmann, , and S. Chakraborty. Security challenges in automotive hardware/software architecture design. In *Proceedings of the Conference on Design, Automation and Test in Europe*, pages 458–463. EDA Consortium, 2013.

[230] A. Sargolzaei, C. D. Crane, A. Abbaspour, and S. Noei. A machine learning approach for fault detection in vehicular cyber-physical systems. In *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 636–640. IEEE, 2016.

[231] A. Saxena, J. Celaya, I. Roychoudhury, S. Saha, B. Saha, and K. Goebel. Designing data-driven battery prognostic approaches for variable loading profiles: Some lessons learned. *International Journal of Energy Research*, 34(2):152–163, 2010.

[232] T. Schütze. Automotive security: Cryptography for car2x communication. In *Embedded World Conference.* Citeseer, 2011.

[233] K. Seiberts and J. Childers. Relay attack prevention for passive entry/passive start systems, Aug. 11 2015. US Patent 9,102,296.

[234] R. Sens. Be ready to fight new 5g vulnerabilities. *Network Security*, 2018(10):6–7, 2018.

[235] D. P. Shepard, J. A. Bhatti, and T. E. Humphreys. Drone hack: Spoofing attack demonstration on a civilian unmanned aerial vehicle. 2012.

[236] D. Shin, M. Poncino, and E. Macii. Thermal Management of Batteries Using a Hybrid Supercapacitor Architecture. *Proceedings of the Conference on Design, Automation & Test in Europe (DATE'14)*, 2014.

[237] D. Shin, M. Poncino, E. Macii, and N. Chang. A statistical model-based cell-to-cell variability management of li-ion battery pack. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 34(2):252–265, 2015.

[238] Y. Shoukry, P. Martin, P. Tabuada, and M. Srivastava. Non-invasive spoofing attacks for anti-lock braking systems. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 55–72. Springer, 2013.

[239] Y. Shoukry, P. Martin, P. Tabuada, and M. Srivastava. Non-invasive spoofing attacks for anti-lock braking systems. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 55–72. Springer, 2013.

[240] Y. Shoukry, S. Mishra, Z. Luo, and S. Diggavi. 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, 2018.

[241] Y. Shoukry, S. Mishra, Z. Luo, and S. Diggavi. 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, 2018.

[242] Y. Shoukry, P. Nuzzo, A. Puggelli, A. L. Sangiovanni-Vincentelli, S. A. Seshia, and P. Tabuada. Secure state estimation for cyber-physical systems under sensor attacks: A satisfiability modulo theory approach. *IEEE Transactions on Automatic Control*, 62(10):4917–4932, 2017.

[243] R. Shrestha, R. Bajracharya, and S. Y. Nam. Blockchain-based message dissemination in vanet. In *2018 IEEE 3rd International Conference on Computing, Communication and Security (ICCCS)*, pages 161–166. IEEE, 2018.

[244] Y. L. Sit, C. Sturm, L. Reichardt, T. Zwick, and W. Wiesbeck. The ofdm joint radar-communication system: An overview. In *Proc. Int. Conf. Advances in Satellite and Space Communications (SPACOMM 2011)*, pages 69–74, 2011.

[245] C. Sitawarin, A. N. Bhagoji, A. Mosenia, M. Chiang, and P. Mittal. Darts: Deceiving autonomous cars with toxic signs. *arXiv preprint arXiv:1802.06430*, 2018.

[246] J. Slay and M. Miller. Lessons learned from the maroochy water breach. In *International conference on critical infrastructure protection*, pages 73–82. Springer, 2007.

[247] C. Smith. *The car hacker's handbook: a guide for the penetration tester.* no starch Press, 2016.

[248] C. Sommer. Veins: Vehicles in network simulation, 2015.

[249] C. Sommer, D. Eckhoff, A. Brummer, D. S. Buse, F. Hagenauer, S. Joerer, and M. Segata. Veins: The open source vehicular network simulation framework. In *Recent Advances in Network Simulation*, pages 215–252. Springer, 2019.

[250] C. Specification. v2. 0. *Common public radio interface (CPRI)*, pages 1–75, 2004.

[251] Y. Sugiyama, M. Fukui, M. Kikuchi, et al. Traffic jams without bottlenecks—experimental evidence for the physical mechanism of the formation of a jam. *New journal of physics*, 10(3):033001, 2008.

[252] G. E. Suh and S. Devadas. Physical unclonable functions for device authentication and secret key generation. *Proceedings of the 44th annual Design Automation Conference (DAC'07)*, pages 9–14, 2007.

[253] I. A. Sumra, J.-L. Ab Manan, and H. Hasbullah. Timing attack in vehicular network. In *Proceedings of the 15th WSEAS International Conference on Computers, World Scientific and Engineering Academy and Society (WSEAS), Corfu Island, Greece*, pages 151–155, 2011.

[254] J. Sun, C. Zhang, and Y. Fang. An id-based framework achieving privacy and non-repudiation in vehicular ad hoc networks. In *MILCOM 2007-IEEE Military Communications Conference*, pages 1–7. IEEE, 2007.

[255] K. A. O. Suzuki, P. Kemper Filho, and J. R. Morrison. Automatic battery replacement system for uavs: Analysis and design. *Journal of Intelligent & Robotic Systems*, 65(1):563–586, 2012.

[256] A. Takanen, J. D. DeMott, and C. Mille. *Fuzzing for Software Security Testing and Quality Assurance* . Artech House, 2008.

[257] C. C. Tan, H. Wang, S. Zhong, and Q. Li. Ibe-lite: a lightweight identity-based cryptography for body sensor networks. *IEEE Transactions on Information Technology in Biomedicine*, 13(6):926–932, 2009.

[258] T. Tanizawa, T. Suzumiya, and K. Ikeda. Cloud-connected battery management system supporting e-mobility. *Fujitsu Sci. Tech. J*, 51(4):27–35, 2015.

[259] A. Taylor, N. Japkowicz, and S. Leblanc. Frequency-based anomaly detection for the automotive can bus. In *WCICSS*, pages 45–49, 2015.

[260] A. N. S. Team. A detailed analysis of the malware responsible for global iot botnets and massive ddos attacks. `https://www.a10networks.com/marketing-comms/white-papers/investigating-mirai/`. (Accessed on 09/19/2019).

[261] M. Tehranipoor and C. Wang. *Introduction to Hardware Security and Trust.* Springer, 2011.

[262] G. Teschl. *Ordinary differential equations and dynamical systems*, volume 140. American Mathematical Society Providence, 2012.

[263] **A. Lopez**, A. V. Malawade, M. A. Al Faruque, S. Boddupalli, and S. Ray. Security of emergent automotive systems: A tutorial introduction and perspectives on practice. *IEEE Design Test*, 36(6):10–38, 2019.

[264] V. L. L. Thing and J. Wu. Autonomous vehicle security: A taxonomy of attacks and defences. *2016 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pages 164–170, 2016.

[265] C. Torralba. Busting the myth: Yes, cell phones can explode. `http://www.androidauthority.com/busting-the-myth-yes-cell-phones-can-explode-42582/`, 2012.

[266] M. Treiber, A. Hennecke, and D. Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical review E*, 62(2):1805, 2000.

[267] G. A. Ubiergo and W.-L. Jin. Mobility and environment improvement of signalized networks through vehicle-to-infrastructure (v2i) communications. *Transportation Research Part C: Emerging Technologies*, 68:70–82, 2016.

[268] I. Ullah and S. U. Rehman. Analysis of black hole attack on manets using different manet routing protocols, 2010.

[269] T. Urbanik, A. Tanaka, B. Lozner, et al. *Signal timing manual.* Transportation Research Board, 2015.

[270] Former simi valley ceo convicted of selling navy knock-off batteries used on subs and aircraft carriers. `https://www.ice.gov/news/releases/former-simi-valley-ceo-convicted-selling-navy-knock-batteries-used-subs-and-aircraft`, October 2015.

[271] M. Uysal, Z. Ghassemlooy, A. Bekkali, A. Kadri, and H. Menouar. Visible light communication for vehicular networking: performance study of a v2v system using a measured headlamp beam pattern model. *IEEE Vehicular Technology Magazine*, 10(4):45–53, 2015.

[272] A. Van Herrewege, D. Singelee, and I. Verbauwhede. Canauth-a simple, backward compatible broadcast authentication protocol for can bus. In *ECRYPT Workshop on Lightweight Cryptography*, volume 2011, 2011.

[273] T. Vanyo and J. Qian. Battery authentication architecture and implementation for portable devices. September 2004.

[274] P. Varaiya. Max pressure control of a network of signalized intersections. *Transportation Research Part C: Emerging Technologies*, 36:177–195, 2013.

[275] A. Varga. Omnet++ http://www. omnetpp. org. *IEEE Network Interactive*, 16(4), 2002.

[276] A. Varga. Omnet++. In *Modeling and tools for network simulation*, pages 35–59. Springer, 2010.

[277] M. Vasek, M. Thornton, and T. Moore. Empirical analysis of denial-of-service attacks in the bitcoin ecosystem. In *International conference on financial cryptography and data security*, pages 57–71. Springer, 2014.

[278] K. Vatanparvar and M. A. Al Faruque. Battery Lifetime-Aware Automotive Climate Control for Electric Vehicles. *Proceedings of the Design Automation Conference (DAC'15)*, pages 1–6, 2015.

[279] K. Vatanparvar and M. A. Al Faruque. Eco-Friendly Automotive Climate Control and Navigation System for Electric Vehicles. *International Conference on Cyber-Physical Systems (ICCPS)*, pages 1–10, 2016.

[280] K. Vatanparvar and M. A. Al Faruque. OTEM: Optimized Thermal and Energy Management for Hybrid Electrical Energy Storage in Electric Vehicles. *Design Automation and Test in Europe (DATE)*, pages 1–6, 2016.

[281] K. Vatanparvar, J. Wan, and M. A. Al Faruque. Battery-Aware Energy-Optimal Electric Vehicle Driving Management. *International Symposium on Low Power Electronics and Design (ISLPED)*, pages 353–358, 2015.

[282] S. Verma, B. Mallick, and P. Verma. Impact of gray hole attack in vanet. In *2015 1st International Conference on Next Generation Computing Technologies (NGCT)*, pages 127–130. IEEE, 2015.

[283] J. Wan, A. Lopez, and M. A. A. Faruque. Physical layer key generation: Securing wireless communication in automotive cyber-physical systems. *ACM Transactions on Cyber-Physical Systems*, 3(2):13, 2018.

[284] J. Wan, A. B. Lopez, and M. A. Al Faruque. Exploiting wireless channel randomness to generate keys for automotive cyber-physical system security. In *2016 ACM/IEEE 7th International Conference on Cyber-Physical Systems (ICCPS)*, pages 1–10. IEEE, 2016.

[285] J. Wan, A. B. Lopez, and M. A. A. Faruque. Exploiting wireless channel randomness to generate keys for automotive cyber-physical system security. In *2016 ACM/IEEE 7th International Conference on Cyber-Physical Systems (ICCPS)*, pages 1–10, April 2016.

[286] S. Weber. City sees red after engineers hack traffic signals: Engineers hacked traffic system as part of a labor protest. 2009.

[287] S. Weintraub. Gogoro scooter and battery swap distribution model get smarter and spread to new cities. `https://electrek.co/2016/01/06/gogoro-scooter-battery-swap/`, 2016.

[288] L. Whitney. Viking horde malware attacks android devices - cnet. `https://www.cnet.com/news/viking-horde-malware-attacks-android-devices/`. (Accessed on 09/19/2019).

[289] M. Wolf. Report of the nsf workshop on internet-of-things (iot) systems. Technical report, November 2019.

[290] M. Wolf, A. Weimerskirch, and C. Paar. Security in automotive bus systems. In *Workshop on Embedded Security in Cars*, 2004.

[291] A. M. Wyglinski, X. Huang, T. Padir, L. Lai, T. R. Eisenbarth, and K. Venkatasubramanian. Security of autonomous systems employing embedded computing and sensors. *IEEE micro*, 33(1):80–86, 2013.

[292] X. Xiong, D. S. Wong, and X. Deng. Tinypairing: A fast and lightweight pairing-based cryptographic library for wireless sensor networks. In *2010 IEEE Wireless Communication and Networking Conference*, pages 1–6. IEEE, 2010.

[293] C. Xu, H. Liu, P. Li, and P. Wang. A remote attestation security model based on privacy-preserving blockchain for v2x. *IEEE Access*, 6:67809–67818, 2018.

[294] C. Yan, W. Xu, and J. Liu. Can you trust autonomous vehicles: Contactless attacks against sensors of self-driving vehicle. *DEF CON*, 24, 2016.

[295] N. Yang, L. Wang, G. Geraci, M. Elkashlan, J. Yuan, and M. Di Renzo. Safeguarding 5g wireless communication networks using physical layer security. *IEEE Communications Magazine*, 53(4):20–27, 2015.

[296] T. Yang, L. Kong, W. Xin, J. Hu, and Z. Chen. Resisting relay attacks on vehicular passive keyless entry and start systems. In *2012 9th International Conference on Fuzzy Systems and Knowledge Discovery*, pages 2232–2236. IEEE, 2012.

[297] C. Ye, S. Mathur, A. Reznik, Y. Shah, W. Trappe, and N. B. Mandayam. Information-theoretically secret key generation for fading wireless channels. *IEEE Transactions on Information Forensics and Security*, pages 240–254, 2010.

[298] C. Yuan, J. Thai, and A. M. Bayen. Zubers against zlyfts apocalypse: An analysis framework for dos attacks on mobility-as-a-service systems. In *Proceedings of the 7th International Conference on Cyber-Physical Systems*, page 24. IEEE Press, 2016.

[299] K. Zaidi, M. Milojevic, V. Rakocevic, and M. Rajarajan. Data-centric rogue node detection in vanets. In *2014 IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications*, pages 398–405. IEEE, 2014.

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

[301] F. Zhang, A. Hennessy, and S. Bhunia. Robust counterfeit pcb detection exploiting intrinsic trace impedance variations. In *2015 IEEE 33rd VLSI Test Symposium (VTS)*, pages 1–6, April 2015.

[302] J. Zhang, F.-Y. Wang, K. Wang, et al. Data-driven intelligent transportation systems: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 12(4):1624–1639, 2011.

[303] T. Zhang, H. Antunes, and S. Aggarwal. Defending connected vehicles against malware: Challenges and a solution framework. *IEEE Internet of Things journal*, 1(1):10–21, 2014.

[304] T. Zhou, R. R. Choudhury, P. Ning, and K. Chakrabarty. P2dap—sybil attacks detection in vehicular ad hoc networks. *IEEE journal on selected areas in communications*, 29(3):582–594, 2011.

[305] T. Ziermann, S. Wildermann, and J. Teich. Can+: A new backward-compatible controller area network (can) protocol with up to 16x higher data rates. In *Proceedings of the Conference on Design, Automation and Test in Europe*, pages 1088–1093. European Design and Automation Association, 2009.