Autonomous Driving Security: State of the Art and Challenges

Cong Gao, Geng Wang, Weisong Shi, *Fellow, IEEE*, Zhongmin Wang, and Yanping Chen

***Abstract*—The autonomous driving industry has mushroomed over the past decade. Although autonomous driving has undoubtedly become one of the most promising technologies of this century, its development faces multiple challenges, of which security is the major concern. In this paper, we present a thorough analysis of autonomous driving security. At first, the attack surface of autonomous driving is presented. After an analysis of the operation of autonomous driving in terms of key components and technologies, the security of autonomous driving is elaborated in four dimensions: sensors, operating system, control system, and vehicle-to-everything communication. Sensor security is examined from five components which are mainly responsible for self-positioning and environmental perception. The analysis of operating system security, the second dimension, is concentrated on the robot operating system. Concerning the control system security, controller area network is approached mainly from vulnerabilities and protection measures. The fourth dimension, vehicle-to-everything communication security, is probed from four categories of attacks: authenticity/identification, availability, data integrity, and confidentiality with corresponding solutions. Moreover, the drawbacks of existing methods adopted in the four dimensions are also provided. Finally, a conceptual multi-layer defense framework is proposed to secure the information flow from external communication to the physical autonomous vehicle.**

***Index Terms*—Autonomous driving, unmanned vehicle, security, attack surface, sensor, robot operating system, data distribution service, control area network, V2X communication.**

I. Introduction

With the rapid improvement of intelligent vehicles, autonomous driving has attracted much research attention. Autonomous vehicles are considered to be beneficial for alleviating traffic congestion and reducing the number of road accidents. However, current autonomous driving technologies are immature and still in development. The safety of the passengers and the vehicle itself are far from guaranteed [1] [2]. For instance, in 2018, an Uber unmanned vehicle collided with a pedestrian wheeling a bicycle across the road during a road test in Arizona [3]. This was the world's first case of an autonomous vehicle accident which caused the death of a pedestrian. The incident subsequently led to a stormy discussion of the safety of autonomous vehicles.

This work was partly supported by the Science and Technology Project of the Shaanxi Provincial Science and Technology Department, China (Grant No. 2019ZDLGY07-08), the Scientific Research Program Funded by Shaanxi Provincial Education Department, China (Grant No. 21JP115), and the Special Funds for Construction of Key Disciplines in Universities in Shaanxi, China. *(Corresponding author: Cong Gao.)*

Cong Gao and Geng Wang are with the School of Computer Science and Technology, Xi'an University of Posts and Telecommunications, Xi'an 710121, China (e-mail: cgao@xupt.edu.cn; gwang_xupt@126.com).

Weisong Shi is with the College of Engineering, Wayne State University, Detroit, MI 48202 USA (e-mail: weisong@wayne.edu).

Zhongmin Wang and Yanping Chen are with the Shaanxi Key Laboratory of Network Data Analysis and Intelligent Processing, Xi'an University of Posts and Telecommunications, Xi'an 710121, China (e-mail: zmwang@xupt.edu.cn; chenyp@xupt.edu.cn).

A. Autonomous Driving Security

An autonomous vehicle is a comprehensive system which mainly consists of a positioning system, a perception system, a planning system, and a control system [4]. The security of autonomous vehicles generally refers to the security during the driving process, including the security of the sensor, operating system, control system, and vehicle-to-everything (V2X) communication.

1) Sensor security: Sensor security mainly deals with the security of the actual components, such as the on-board sensors and on-board chips. For instance, Google's self-driving vehicles employ a variety of sensors to detect the driving environment. The collected sensor data is used to analyze whether a vehicle is in a safe driving state.

2) Operating system security: Operating system security refers to ensuring the integrity and availability of the operating system and preventing unauthorized access. At present, most autonomous vehicles are developed based on a robot system. For instance, Baidu's autonomous vehicle platform Apollo [5] is based on the most famous robot operating system, ROS [6]. ROS is a robot middleware platform which provides the basic functions of an operating system for heterogeneous computer clusters. However, ROS was originally designed without considering security. Other similar operating systems also suffer from this problem.

3) Control system security: Control system security guarantees that the on-board decision-making system gives correct instructions for steering, acceleration, deceleration, and parking of the autonomous vehicle based on the data collected from both the environment and the vehicle itself. However, with the increasing variety of external interfaces of a vehicle, novel attack surfaces keep emerging. Thus, the control system is vulnerable to illegal invasions.

4) V2X communication security: V2X communication security refers to the security of the communication of vehicle-to-vehicle (V2V), vehicle-to-infrastructure (V2I), vehicle-to-pedestrian (V2P), and vehicle-to-network (V2N). The design of a vehicle network system is supposed to guarantee the above communication against attacks. Moreover, the information about surrounding vehicles and environmental conditions

coming from V2X communication further contributes to the security of a vehicle.

B. Attack Surface

The notion of *attack surface* usually attributed to Michael Howard of Microsoft. It is informally introduced to act as an indicator of the security of a software system [7].

Early research on attack surface [8] [9] [10] [11] [12] mainly focused on software systems and laid a solid foundation for subsequent study. Michael Howard considered that attack surface is a set of attack features: open sockets, open RPC endpoints, open named pipes, and services, etc [7]. Manadhata *et al*. [12] presented the definition that a system's attack surface is the subset of resources that an attacker can use to attack the system.

Ren *et al*. [2] briefly categorized security threats surrounding an autonomous vehicle into three groups of attacks surfaces: various sensors, in-vehicle access and control systems, and in-vehicle network protocols. Each group was described in terms of attack types and defense strategies.

Recent literature about attack surface focused on creating empirical and theoretical measures for the attack surface of a software system or computer network [13], such as [14] [15] [16] [17].

In the field of autonomous driving, notable literatures concerning attack surface are as follows.

Maple *et al*. [18] developed a reference architecture using a hybrid functional-communication viewpoint for attack surface analysis of connected autonomous vehicles (CAVs). Devices, edge and cloud systems which CAVs interact with were discussed. The technique attack tree is used to analyze attacked functions, attack surfaces, attacked assets, and attacker capability.

Salfer *et al*. [19] proposed a method for the attack surface and vulnerability assessment automation of automotive electronic control units (ECUs) based on development data and software flash images. The attack surface includes internal communication interfaces, external/user-accessible interfaces, and low-level hardware interfaces.

Checkoway *et al*. [20] conducted a detailed analysis of the external attack surface for automobiles. This work mainly focused on remote compromise. Four types of automotive threat models were described, including direct physical access, indirect physical access, short-range wireless access, and long-range wireless.

In [21], threat areas of in-vehicle infotainment systems were discussed. Vulnerabilities and possible mitigation strategies were presented. Seven vulnerabilities of Linux-based in-vehicle infotainment systems and fifteen potential attack surfaces were identified.

Chattopadhyay *et al*. [22] developed a security-by-design framework for autonomous vehicles. The framework contains a high-level model which defines the attack surfaces of autonomous vehicles into three layers: 1) the core layer defined by the physical enclosure of an autonomous vehicle, 2) the interface/gateway layer characterized by the collection of connectivity interfaces between an autonomous vehicle and the external world, and 3) the infrastructure layer composed of all the infrastructure and backend modules which are trusted by and connected to an autonomous vehicle.

Dominic *et al*. [23] presented a risk assessment framework for autonomous and cooperative automated driving. A threat model was proposed based on the threat model described by the national highway traffic safety administration (NHTSA) [24] and security requirements described by the E-safety vehicle intrusion protected applications (EVITA) project [25]. Attack surfaces were described in five categories: inertial/odometric, range sensors, global positioning system (GPS), map update, and V2V/V2I.

Petit *et al*. [26] studied the potential cyber attacks against automated vehicles. The attack surfaces in autonomous automated vehicles and cooperative automated vehicles were analyzed, respectively. The analysis was conducted in the following aspects: target, means, feasibility of attack, physical access, ease of detection by driver/system, probability of success, direct consequences, hazard created, and mitigation techniques.

Based on the analysis of the above literatures, we broadly divide the attack surfaces of autonomous driving into three categories. As shown in Fig. 1, they are sensors, in-vehicle systems, and V2X. For sensors: GNSS/IMU stands for global navigation satellite system and inertial measurement unit. LiDAR is short for light detection and ranging. For in-vehicle systems: OBD-II is short for the second generation of on-board diagnostics. TPMS stands for tire pressure monitoring system. ADAS is short for advanced driving assistance system. For V2X: OTA stands for over-the-air. It is essentially just a synonym for wireless. DSRC is short for dedicated short range communication. Fig. 1 is by no means exhaustive but aims to raise the security issues of autonomous vehicles.

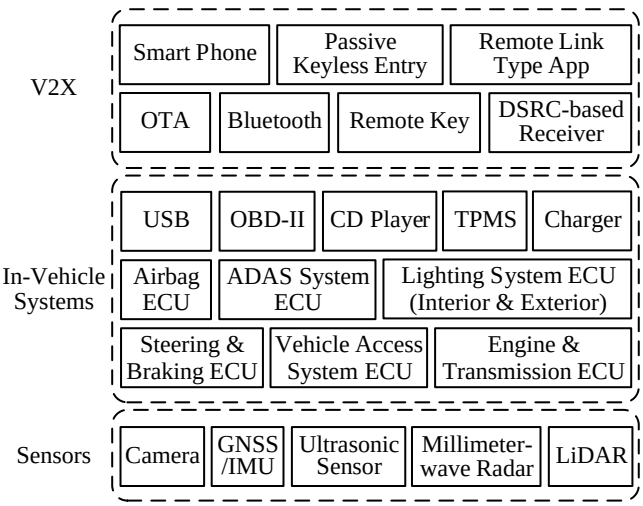

Fig. 1. Attack surfaces of autonomous driving.

C. Content and Roadmap

In this paper, we review the state of the art and challenges involving the above four aspects of autonomous driving and point out the drawbacks of existing solutions. The main

components and related technologies of autonomous driving are presented. The discussion of sensor security is focused on the cameras, GNSS/IMUs, ultrasonic sensors, millimeter-wave radar, and LiDAR. The discussion of operating system security is focused on ROS. A security enhancement data distribution service adopted by ROS version 2 is described in detail. The analysis of control system security is focused on the controller area network (CAN). The vulnerabilities of CAN are analyzed based on five attack paths: OBD-II, electronic vehicle charger, CD player, TPMS, and Bluetooth. Two types of protection methods are presented: those based on encryption/authentication and those based on intrusion detection. Recent development of the control area network standard is also presented based on CAN with Flexible Data-rate (CAN FD). V2X communication security is analyzed based on four categories of attacks: authenticity/identification, availability, data integrity, and confidentiality. Moreover, the blockchain-based security measures for vehicular network are reviewed. At last, six real-world security incidents of autonomous vehicles are presented. Then, a conceptual multi-layer defense framework for the security of autonomous driving is proposed.

The rest of this article is structured as follows. In Section II, we review the main components and technologies of an autonomous driving system. In Section III, we discuss the security of five key sensors for autonomous vehicles. In Section IV, we analyze the security of the popular operating systems for autonomous vehicles. The discussion is concentrated on ROS, which plays a dominant role in the field of autonomous driving. In Section V, we discuss the security of control systems based on CAN. Vulnerabilities, attacks, and protections of CAN are presented. New standard of CAN is presented based on CAN FD. In Section VI, we summarize attacks against the communication in the Internet of Vehicles (IoVs) and the corresponding solutions. In Section VII-A, six real-world security incidents of autonomous vehicles are introduced. These incidents are presented in four categories: sensor security, operating system security, control system security, and V2X communication security. In Section VII-B, we propose a conceptual defense framework for automotive information security. Finally, we present our conclusions in Section VIII.

II. Autonomous Driving Technologies

An autonomous driving system is a kind of intelligent system that realizes autonomous driving based on on-board computer systems. It is an integration of multiple technologies. Generally speaking, an autonomous driving system requires powerful computing ability. The computing resources are responsible for the realization of the vehicle positioning, environmental perception, path planning, motion control, etc. For instance, Xiao *et al*. [27] proposed a blockchain-based algorithm called DAER to allocate resources for intensive computing tasks. In general, the realization of an autonomous driving system is based on multi-sensor information fusion and should meet the requirements of high performance and high security. The security of the related technologies for autonomous driving is a prerequisite for ensuring the security of autonomous vehicles on the road.

The autonomous driving technology stack is shown in Fig. 2. There are two major aspects: components and technologies.

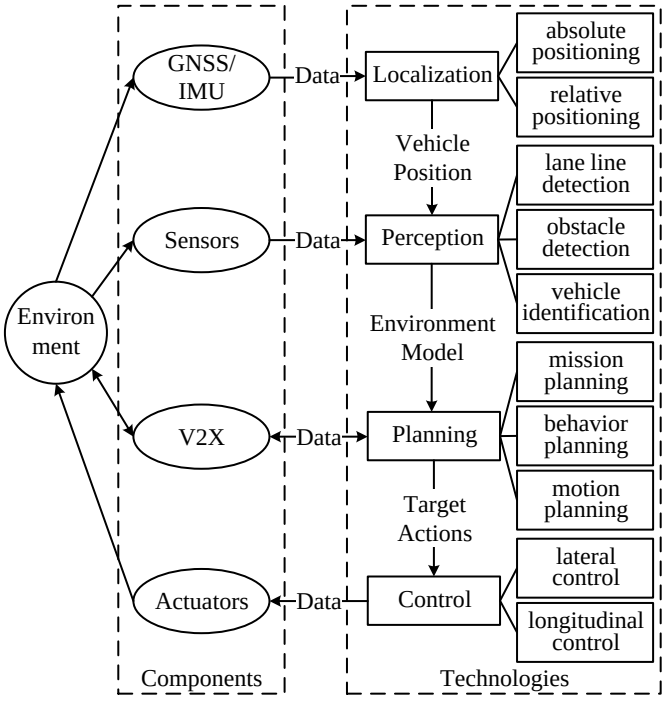

Fig. 2. Technology stack of autonomous driving.

A. Components

The key components of autonomous driving include GNSS/IMUs, sensors, V2X, and actuators. The GNSS/IMU is critical in the localization. It is a core component for sensor fusion and safe driving. Sensors play a pivotal role in environmental perception. Therefore, sensors should be deployed around an autonomous vehicle. The detection coverage of cooperative homogeneous sensors is often made to be overlapping so as to provide redundancy and accuracy. Different sensors use different detection technologies to perceive specific environmental information. An environment model is built based on this information. For instance, V2X is able to collect real-time information about the surrounding vehicles and environmental conditions. This information is used for planning, which is critical in reducing traffic jams and enhancing the safety of the driving. Target actions given by the planning process are based on the information related to V2X and the model of the environment. The control module issues commands, in accordance with the actions aimed at, to the corresponding actuators. An actuator acts on the environment and changes the status of the vehicle. The technologies mainly involve localization, perception, planning, and control.

B. Technologies

1) Localization: Existing solutions to the localization of autonomous vehicles fall into two groups: 1) a vehicle networking solution based on V2X with shared location information and 2) a single agent solution based on multi-sensor

information fusion. To ensure the safe and reliable operation of autonomous vehicles on the road, the accurate positioning of the vehicles is a prerequisite.

As one of the core functions of vehicle sensing systems, positioning plays an extremely important role in research into autonomous vehicles. In other words, positioning is a fundamental problem in this research area. The GNSS/IMU package is an effective solution for positioning of autonomous vehicles [28]. However, this method is unable to achieve high-precision positioning when the GNSS signals are weak, such as in underground parking lots and urban areas surrounded with high-rise buildings. Besides, GNSS signals are easily interfered with by a GPS jammer [29]. Map-assisted positioning is another popular type of autonomous vehicle positioning method. Simultaneous localization and mapping (SLAM) [30] is an example of this kind of algorithm. This technology is also known as concurrent mapping and localization (CML). SLAM determines the current position of a vehicle based on the observed environmental characteristics. However, during a long-distance movement, the deviation of the SLAM positioning gradually increases, thus resulting in an inaccurate positioning, which is unacceptable for certain application scenarios. The above problem with SLAM positioning is effectively addressed by employing light detection and ranging (LiDAR) to construct a point cloud map of the area of interest in advance [31] [32]. Several *semantics* are added to the map, both automatically and manually, such as specific markings of the lane lines, the location of traffic lights, and traffic rules on different roads. This kind of semantic map is called a high definition (HD) map.

2) Perception: As the most challenging module in autonomous vehicles, perception system directly affects the results given by planning system and control system. Conventional perception modules mainly utilize computer vision technologies to extract information of the driving environment. The obtained information is used to conduct lane lines detection, obstacle detection, and vehicle recognition/tracking, etc.

Autonomous vehicles are equipped with a variety of sensors. Among these sensors, ultrasonic radar, millimeter-wave radar, LiDAR, and cameras can be considered as *vision* in a broad sense. Due to low response speed and low resolution, ultrasonic radars are typically used for coarse-grained occasions, such as car reversing aid alarm systems [33]. On the one hand, when a vehicle is running at a high speed, the performance of ultrasonic radar ranging is unable to catch up with the variation of displacement. On the other hand, as the scattering angle of an ultrasonic radar is large, the signal reflected back is weak especially for the measurement of a distant target. Hence, the decrease in measurement accuracy might be significant. Millimeter-wave radar and LiDAR are mainly responsible for the ranging of medium and long distances. LiDAR generally relies on multiple laser transmitters and receivers to build three-dimensional point cloud maps. These maps are used to achieve real-time environmental perception. Two distinct advantages of LiDAR ranging are high-precision and long-distance. However, the actual performance of LiDAR might be poor in certain weather conditions (e.g., rain, snow, and fog), since the straight laser is blocked by obstacles. A millimeter-wave radar emits radio waves to determine the position of a target. This kind of radar is hardly affected by harsh weather conditions, thus it is better than LiDAR in this respect. However, millimeter-wave radars are less capable in describing the shape of an object than that of LiDAR. Cameras are mainly used for capturing the information about traffic lights, traffic signs, and other objects. In general, the images collected by a camera are examined and partitioned to extract key features involving potential objects of interest. The extracted information is then compared with a feature library for the purpose of image recognition. However, the functionality of a camera is dramatically crippled by strong light or bad weather.

Object recognition and tracking are two important goals of the perception module [34]. At present, the implementation of object recognition is mainly based on convolution neural networks (CNNs) [35]. For object tracking, deep learning technologies also have advantages over the conventional computer vision technologies [36].

3) Planning: The planning module of an autonomous vehicle can be divided into three layers: mission planning [37], behavior planning [38], and motion planning [39]. In most cases, they are conducted in the sequential order shown in Fig. 3.

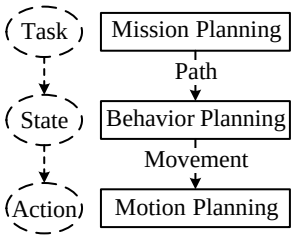

Fig. 3. The three layers of planning.

- *Mission planning*. Mission planning is also referred to as path planning or routing planning. It focuses on the task-level planning, such as the selection of a path between a starting point and an end point [40]. A given road system can be considered as a weighted directed diagram. This diagram contains plenty of information, such as the connectivity among the different roads, traffic rules, and the widths of the roads. This information contributes the *semantics* of an HD map mentioned in Section II-B1. As each directed edge in the diagram is weighted, the core idea of path planning for an autonomous vehicle is essentially the path search problem in a weighted directed diagram. In order to make a vehicle move from A to B, it is expected to obtain an optimal path which is subject to several constraints, such as time, distance, and congestion.
- *Behavior planning*. Behavior planning is also called decision making. Since autonomous vehicles usually travel in a complicated environment which is full of uncertainty and dynamics, challenges may come from 1) the degradation of the performance of the sensors and actuators, such as a snow-covered LiDAR and a skidding tire on

wet ground, 2) vehicles and pedestrians breaking the rules, or other objects, such as reckless animals and boxes falling off a truck, and 3) unknown social conventions in unfamiliar areas, such as local festivals and gatherings. Therefore, behavior planning is introduced to make the appropriate decisions for the next move of the autonomous vehicle, according to the result of the mission planning and a wide variety of live information. For instance, behavior planning instructs the vehicle to follow or pass other vehicles, wait for or pass by pedestrians, etc. One approach to behavior planning is to use a complex finite state machine (FSM) which contains a large number of actions [41] [42]. The FSM starts from an initial state and jumps to different states based on the variations of the driving scenario. The corresponding actions are passed to the motion planning.

- *Motion planning*. Motion planning refers to the process of planning a series of consecutive actions. This series corresponds to a specific goal, such as acceleration and obstacle avoidance. In general, there are two important performance metrics for a motion planning algorithm: computational efficiency and integrity [40]. Computational efficiency refers to the processing speed of accomplishing a motion plan. The computational efficiency of a motion planning algorithm depends largely on the corresponding configuration space. The integrity of a motion planning algorithm is described as follows. Provided a problem is solvable, the motion planning algorithm is able to find a solution in bounded time. For an unsolvable problem, the algorithm is capable of justifying its infeasibility. In the scenario of autonomous driving, the initial configuration of a motion planning algorithm usually contains the current states of the vehicle, including its position, linear velocity, angular velocity, etc. The target configuration is derived from the behavior planning. In practice, the movement of a vehicle always possesses certain restrictions, such as maximum steering angle, maximum acceleration, and maximum speed. These constraints are defined in the configuration space.

4) Control: When an autonomous vehicle completes its self-positioning and its perception of its surrounding environment, as well as its planning decision, it needs to transform the obtained series of action into controlled operations of the vehicle. In general, vehicle control consists of lateral control and longitudinal control [43]. Lateral control refers to the adjustment of the steering wheel and the tires' lateral force. Longitudinal control refers to the acceleration and braking of the vehicle.

In practice, the most common demands for control of an autonomous vehicle are acceleration, steering, and braking. The input of the control module is a series of path points. The role of the control module is to make the vehicle move along these path points to the greatest extent possible. A good control module should possess three features: accuracy, feasibility, and stability. Feedback control is widely used in the field of automation control. The most typical feedback controller is the proportional–integral–derivative (PID) controller [44]. As a linear controller, ordinary PID controllers are widely used in industrial processes due to their simplicity. A PID controller is a feedback control model based on error signals. Here, the error signals consist of three parts: proportion error, integral error, and derivative error. The idea of PID control possesses three advantages: simplicity, robustness, and reliability. However, the application of a PID controller to autonomous vehicles faces the following challenge: the algorithms of a PID controller need to determine specific hyper-parameters and their values [45]. For autonomous driving, the uncertainty of the external environment and the non-holonomic constraints of a vehicle make it difficult to find the appropriate hyper-parameters and their corresponding optimal values.

5) Computing system: As the computing resources available to the on-board computing units are limited, it is difficult to deploy a large number of computation-intensive services on the vehicle. Edge computing is an effective way to address this problem. Zhang *et al*. [46] proposed a vehicular data analysis platform called OpenVDAP. The platform includes four main parts: 1) an on-board heterogeneous vehicle computing/communication unit (VCU), 2) an isolation-supported and security/privacy-aware vehicle operating system (EdgeOSV), 3) a driving data integrator (DDI), and 4) an edge-aware application library (LibvDAP). This platform is deployed on the autonomous vehicle to perform the calculations for the on-board applications. The service quality of the on-board applications and user experience are improved. Liu *et al*. [47] summarized the most advanced autonomous driving computing systems. There are seven performance indicators, nine key technologies, and twelve challenges.

III. Sensor Security

Autonomous vehicles are equipped with a variety of sensors, such as camera, GNSS/IMU, ultrasonic radar, millimeter-wave radar, and LiDAR. These sensors are responsible for collecting information about the positioning of the vehicle itself, its surrounding environment, etc.

El-Rewini *et al*. [48] presented a comprehensive review of potential cyber threats related to the sensing layer. Sensors of autonomous vehicles were classified as two categories: vehicle dynamics sensors (e.g., TPMSs, magnetic encoders, and inertial sensors) and environment sensors (e.g., LiDAR, ultrasonic sensors, cameras, radio detection and ranging systems, and GPS units). The authors also offered perspectives through existing countermeasures from literature and stressed the need for data-driven cybersecurity solutions.

Sensors are at the forefront of the field of autonomous driving. At present, most attacks against autonomous vehicles are related to sensors. Common attacks carried out against sensors inject misinformation or try to degrade the performance of the sensors by any means possible. As different sensors possess different operating principles, various types of attacks are used [26].

A. Camera

1) Role in autonomous driving: As computer vision assists autonomous vehicles to complete many perception tasks, the

camera is the most basic vision sensor, and is indispensable for autonomous driving [49]. Cameras used by autonomous vehicles are mainly divided into three categories: monocular cameras, binocular cameras, and multinocular cameras. The monocular camera is widely used in ADASs. However, there is a drawback to the use of a monocular camera. For a monocular camera with fixed resolution, a farther scene corresponds to a larger view, but it will be less clear. In contrast, a closer scene appears more clear. Although the binocular camera addresses the above problem of monocular camera, monocular cameras are used more than binocular cameras in autonomous driving at present. The main reasons are the expensive computational overhead of binocular camera algorithms and the shortage of space in an autonomous vehicle for such equipment.

2) Attacks and countermeasures: In general, autonomous vehicles of Level 3/4 require the cooperation of multiple cameras for the perception of the surrounding environment, including pedestrians, lane lines, traffic signs, other vehicles, etc. In the task of traffic light recognition, if cameras capture a red light or a pedestrian, the vehicle should slow down or stop to avoid an accident. Hackers can place extra traffic lights or fake pedestrians to trigger a stop of the vehicle. In addition, a highlighted IR laser can also interfere with cameras, preventing the generation of effective images [50]. Attacks against camera and underlying computer vision algorithms of autonomous vehicles are common [50] [51].

Zhang *et al*. [52] proposed a framework based on three cameras to detect attacks against cameras. This framework uses the information captured by the cameras to obtain different versions of depth maps (i.e., disparity). The distribution of disparity errors is subsequently analyzed to detect attacks.

Cao *et al*. [53] pointed out that all prior studies on autonomous driving systems only focused on camera or LiDAR-based autonomous driving perception alone. The authors studied the security of multi-sensor fusion based perception in autonomous driving. A novel attack pipeline was developed to attack all fusion sources simultaneously. The authors succeeded in generating a physically-realizable, adversarial 3D-printed object that comprises both camera and LiDAR. In addition, corresponding defense strategies were also discussed.

DiPalma *et al*. [54] developed an adversarial patch attack against camera-based obstacle detection. The adversarial patch with appropriate size and appearance is added to the back of a box truck. A victim autonomous vehicle is expected to be unaware of the box truck and thus collide into it. Experiments of the attack were conducted against an Apollo autonomous vehicle running in production-grade autonomous driving simulator LGSVL [55].

Kyrkou *et al*. [56] pointed out that advanced artificial intelligence and machine learning techniques play an vital role in proactive defense against attacks on autonomous vehicles' cameras. The authors developed a project called CARAMEL [57]. This project shows the use of AI/ML-based techniques in detection and possibly mitigation of dynamic cyber-attacks on the camera system/data in autonomous driving. Both external attacks on camera sensor and direct attacks on camera sensor data were analyzed. Experiments were carried out on CARLA [58].

B. GNSS/IMU

1) Role in autonomous driving: GNSS/IMU is a real-time localization method in autonomous driving [59]. As a highly accurate localization method, GNSS-RTK is able to achieve centimeter-level position accuracy under dynamic measurement. Here, RTK stands for real-time kinematics. However, the frequency of location update is low, and the satellite signal can be easily blocked [60]. IMUs and odometers are used to accumulate displacement and direction variations for the purpose of compensation during the period between two consecutive positionings of the GNSS-RTK. Although the update frequency is high for the IMU and odometer, there are accumulated errors. Through the combination of GNSS and IMU, we can achieve real-time localization with low delay, high precision, and high frequency.

2) Attacks and countermeasures: When a high-powered fake GPS signal transmitter is placed near an autonomous vehicle, the genuine GPS signal might be covered up. Thus, the localization of the autonomous vehicle is misled [61]. By combining two simple attack methods, GNSS signal jamming and spoofing, GNSS/IMU localization can be easily compromised [62].

Magiera *et al*. [63] proposed a spoofing detection method using phase delay measurement. This method uses multiple antennas to receive GPS signals of different qualities, then the accuracy and precision of the phase delay estimation are assessed.

In order to eliminate spoofing signals, Han *et al*. [64] constructed the subspace projection of the spoofing signals using the pseudo-code characteristics of spoofing signals.

Dasgupta *et al*. [65] proposed a prediction-based spoofing attack detection scheme with the long short-term memory (LSTM) model. The distance between two consecutive locations of an autonomous vehicle is predicted by the LSTM model. Based on this distance, a threshold value is obtained with the positioning error of the GNSS module and prediction error. Experiments were conducted with a real-world driving dataset called Comma2k19 [66].

Mit *et al*. [67] analyzed Tesla's Level 2 autonomous driving system under different GNSS spoofing scenarios. Experiments covered different combinations of GNSS attacks. To examine various multi-constellation mitigation, GPS was spoofed and other constellations were jammed. The evaluation of a self-developed software library which detects attacks and provides authentication and protection of GNSS was also incorporated.

Dasgupta *et al*. [68] developed a deep reinforcement learning (RL)-based turn-by-turn GNSS spoofing attack detection using low-cost in-vehicle sensor data. Experiments were carried out with the Honda Research Institute Driving Dataset [69]. The dataset contains information of various vehicle sensor data for suburban and urban driving scenarios. The dataset was used to generate attack and non-attack datasets, develop a deep RL model, and evaluate the performance of the RL-based attack detection model. The threshold value obtained from the trained RL model is compared with a differential distance which is calculated using real-time GNSS data. If the differential distance is greater than the threshold, an attack is detected; otherwise, no attack is detected.

Broumandan *et al*. [70] proposed a spoofing detection model based on consistency check between GNSS and IMU/odometer package. This model focuses on the utilization of inertial measurement units and vehicle odometer readings. A spoofing attack is detected by analyzing GNSS and IMU/odometer measurments independently during an observation period.

Song *et al*. [71] developed a credible navigation algorithm for GNSS attack detection using an auxiliary sensor system. The auxiliary sensor system is on mobile terminals. A credible Kalman filter and measurement information given by the auxiliary sensor system are used to verify the credibility of the GNSS positioning result. A credible verification window and a credible verification threshold are used to detect different kinds of GNSS attacks, including GNSS jump attacks and GNSS slow-change attacks.

C. Ultrasonic Sensor

1) Role in autonomous driving: Ultrasonic sensors were first introduced into vehicles for automated parking assistance systems [72]. An ultrasonic sensor emits an ultrasonic signal in a certain direction through ultrasonic transmitting devices. A timer starts at the moment the signal is transmitted. The emitted ultrasonic signal is reflected back when it encounters obstacles during the transmission. When the reflected signal is received by the corresponding receiver, the timer stops. Based on the recorded time interval, the distance between the vehicle and the obstacle can be calculated.

2) Attacks and countermeasures: Attacks threatening ultrasonic sensors mainly include spoofing attacks and jamming attacks.

Xu *et al*. [33] developed random spoofing, adaptive spoofing, and jamming attacks on ultrasonic sensors and validated these attacks on stand-alone sensors and moving vehicles. Experimental results showed that blindness/malfunction of sensors/autonomous vehicles occurred. In addition, two defense strategies single-sensor-based physical shift authentication and multiple sensor consistency check were proposed.

Yan *et al*. [73] conducted an actual experiment with a spoofing attack in which an ultrasonic signal generated by hackers was introduced (Section 5 in [73]). The generated signal is designed to reach the receiver of the vehicle earlier than the genuine signal expected to be reflected back. If this is the case, the readings from the vehicle's ultrasonic sensor are falsified. In addition, a jamming attack which aims at reducing the signal-noise ratio (SNR) of the ultrasonic sensor by emitting a continuous ultrasonic signal was also discussed.

Lim *et al*. [74] conducted an in-depth evaluation of vulnerabilities of ultrasonic sensor for autonomous vehicles. Several experimental attacks against ultrasonic sensor are launched. The corresponding impact was analyzed in terms of blind-spot range, coverage of ultrasonic transmitter and receiver, obstacle material, and third-party ultrasonic interference. Countermeasures and mitigation strategies were also briefly introduced.

Lou *et al*. [75] thoroughly studied the signal injection attacks and proposed a physical-layer defense system (SoundFence) to secure ultrasonic sensors in autonomous vehicles. The system verifies the benign measurement results and detects signal injection attacks by analyzing sensor readings and the physical-layer signatures of ultrasonic signals.

D. Millimeter-wave Radar

1) Role in autonomous driving: Millimeter wave generally refers to an electromagnetic wave with a wavelength of 1$\sim$10 mm. In most countries, vehicle-mounted millimeter-wave radar operates in the frequency bands of 24 GHz and 77 GHz [76]. In addition, a few countries have adopted the frequency band of 60 GHz (e.g., Japan). Millimeter-wave is able to work in rainy, foggy, and snowy weather conditions due to its strong penetrating ability.

2) Attacks and countermeasures: If a hacker obtains the waveform parameters of a millimeter wave, a millimeter-wave radar at the same frequency band may be jammed [33]. Moreover, millimetre wave may also be subject to electromagnetic interference.

Yan *et al*. [73] conducted security experiments on the radar and autopilot system in Tesla Model S (Section 6 in [73]). Experimental results showed that millimeter-wave radar of an autonomous vehicle suffers from electromagnetic jamming and spoofing. Multiple sensors for redundancy check were suggested to secure the sensor data. The authors also proposed that randomness should be introduced into control parameters, taking logic check, confidence priority, and attack detection system into consideration when designing a sensor data fusion strategy.

Kapoor *et al*. [77] proposed a spatio-temporal challenge–response (STCR) method. This method emits probing signals in multiple randomly selected directions at the same time. Then, the reflected signals are verified according to their directions of emission and arrival. Hence, suspicious signals are filtered out.

Digital radio frequency memory (DRFM) [78] is a kind of microwave signal storage system, which is characterized by using a digital form to store the signals. Hackers use DRFM to receive the millimeter-wave radar signal of a vehicle and then store several duplicates in a digital memory. These signals are retransmitted later as needed. The transmitting radar of the vehicle is unable to distinguish between a legitimate reflected signal and a signal processed by the hacker [26].

Guan *et al*. [79] proposed an anti-jamming method based on hash functions. The experimental results showed that the method is significantly effective in suppressing the echo interference.

Sun *et al*. [80] conducted an end-to-end security analysis of a millimeter-wave-based sensing system in autonomous vehicles. Practical physical layer attacks and defense strategies were implemented. Five real-world attack scenarios were constructed to spoof a victim autonomous vehicle. A challenge-response authentication scheme and an RF fingerprinting scheme were implemented to detect millimeter-wave-based spoofing attacks.

E. LiDAR

1) Role in autonomous driving: LiDAR is currently the most important sensor for autonomous driving. The operating

principle of LiDAR is to emit a laser beam and receive signals reflected back from a target. Several pieces of information related to the target can be obtained by comparing the outgoing and incoming signals, such as distance, azimuth, altitude, and even shape. LiDAR generates HD maps by capturing dense 3D point cloud data from stationary and moving objects around itself. The advantages of LiDAR lie in its long detection range and accurate describing ability for three-dimensional information of objects.

2) Attacks and countermeasures: Like the above mentioned four sensors, LiDAR can also be easily interfered with. The main ways to attack LiDAR are the spoofing attack and the relay attack. Spoofing attack refers to injecting signals into the LiDAR receivers of the target vehicles, while the relay attack refers to using a transmitter and receiver to inject and receive the signals of the target vehicles, respectively.

Shin *et al*. [81] used a delay component to delay the LiDAR signals returned from a target vehicle. The delayed signals are emitted to the target vehicle by a malicious transmitter.

Cao *et al*. [82] showed two types of attacks: 1) an attack device placed at the roadside emits malicious laser pulses at passing autonomous vehicles and 2) an attack device carried by a vehicle emits malicious laser pulses at nearby victim vehicles. Both these attacks interfere with the LiDAR measurements of the victim vehicles.

Petit *et al*. [50] used two transceivers to relay LiDAR signals from the target vehicle to another vehicle at a different location. In this case, the latter vehicle could receive echoes from the location, thus resulting in false echoes.

Sun *et al*. [83] proposed CARLO to mitigate spoofing attacks on LiDAR. CARLO uses ignored occlusion patterns in the LiDAR point clouds as invariant physical features. Experimental results showed that this method can reduce the average success rate of attacks to 5.5%. They also proposed a general architecture for LiDAR-based perception, which embeds ignored physical features into end-to-end learning. Experimental results showed that the average success rate of attacks was further reduced to about 2.3%.

Changalvala *et al*. [84] developed a 3D quantization index modulation (QIM) data hiding technique for the purpose of securing the raw data from LiDAR sensor. The technique detects tampering of LiDAR sensor data and locates the tampered region. Experiments conducted on the KITTI object detection benchmark suite [85] showed that the proposed method was able to detect and localize insider data tampering attacks.

Yang *et al*. [86] proposed an adversarial attack against deep learning models which perform object detection on raw 3D points collected by a LiDAR sensor of an autonomous vehicle. The proposed attack creates robust adversarial objects which are able to cause behavioral reactions of autonomous driving systems. Defense methods against the above adversarial objects were also discussed.

You *et al*. [87] developed a general methodology called 3D temporal consistency check (3D-TC2). It takes advantage of spatio-temporal information from motion prediction to verify objects detected by 3D object detectors. Experimental results showed that the overall performance of LiDAR spoofing attack detection was satisfactory.

F. Multi-sensor Cross Validation

When observations from several different sensors are combined, there is a robust and comprehensive perception model for autonomous vehicles. In general, for the above five types of sensor, it is easy to attack an individual sensor. However, attacking all the sensors of an autonomous vehicle at the same time becomes more difficult. Currently, production autonomous driving systems predominantly adopt a multi-sensor fusion (MSF) based design, which in principle can be more robust against attacks under the assumption that not all fusion sources are (or can be) attacked at the same time [53]. Thus, it is expected that multi-sensor fusion technologies can effectively mitigate sensor attacks on autonomous vehicles. When the information coming from different sources is inconsistent, the vehicle might be under attack. For example, when GNSS/IMU and LiDAR yield different positioning results, at least one of the two systems might have been attacked. Besides, if a sensor system of an autonomous vehicle believes there is a traffic light, but HD Map indicates there is no traffic light at the same position, then in most cases the sensors of the vehicle are likely to have been attacked.

G. Sensor Failure

On-board vehicle sensors may fail due to bad calibration, erroneous readings, physical or electrical failure, etc. Besides being caused by attacks, abnormal sensor readings may also be caused by failure. However, there is no standard or universally agreed definition for sensor failure [88].

Realpe *et al*. [89] proposed a system called the fault tolerant perception paradigm for fault detection of sensors in autonomous vehicles. The system deals with possible sensor failure by defining a federated data fusion architecture. This architecture contains three modules: 1) object detection, 2) local fusion, 3) master fusion and the fault detection and diagnosis (FDD). Specifically, the master fusion module is used as a reference, then the information is verified by various local filters. Each local filter processes the compatible data between the reference sensor and the data provided by other sensors. FDD estimates the residual values with a support vector machine to determine whether a sensor is faulty based on the differences between the local fusion module and the master fusion module. Experimental results show that the system successfully detects early failure of a single sensor and minimizes the impact of the faulty sensor.

Pous *et al*. [90] used analytical redundancy and a nonlinear transformation to generate residual signals for detection of faulty sensors. The method uses statistical tools to optimally determine a threshold based on the characteristics of the signal, prior probabilities, and other information. The fault detection is performed by comparing the residual and the threshold. This proposal was tested on the pro-SiVIC simulation platform. Experimental results show that this method effectively detects sensor failure.

Byun *et al*. [91] proposed a fault diagnosis logic and signal restoration algorithm. The algorithm converts the speed of each wheel into information about the vehicle's central axis. A reference central axis speed is selected based on this information. The obtained central axis speed is used to estimate the speeds of all the wheels. Then, the estimated speeds are compared with the actual in-vehicle speeds, so as to identify any faults. The premise of this method is that only one sensor fails at any given time. Experimental results show that the proposed algorithm is able to meet the requirements of control performance when one sensor fails.

H. Actual Sensor Failure vs. Attacks

Both actual sensor failure and attacks might lead to wrong decisions in autonomous driving. Moreover, certain attacks are designed in an oversimplified and crude way. They simply aim to cause sensor failure. However, actual sensor failure and attacks against sensors are different.

In most cases, attacks against sensors tend to proceed stealthily. The tampering of sensor data is often mild and not obvious. The tampered sensor data just seems like the normal data. Besides, the expected attack effect is to fool the high-level algorithms by tampering the sensor data. On the contrary, actual sensor failure often results in obvious changes of sensor data, such as no readings for a significant time period, extremely high or low readings. For a multi-sensor fusion system like autonomous driving system, actual sensor failure can be easily noticed by multi-sensor cross validation. In this case, safety measures can be taken timely. Thus, security and safety issues are likely to be prevented. On the contrary, as attacks are hard to be detected, both capacity-constrained artificial intelligence packages equipped with autonomous driving system and a negligent human driver will not be aware an attack until serious incidents happen (e.g., a traffic accident). To the best of our knowledge, there is no literature concerning distinguishing between actual sensor failure and attacks. Researchers tend to study methodologies and techniques to discover and defense attacks. Actual sensor failure is left as hardware problems. Though actual sensor failure also leads to abnormal sensor data, it is often neglected by researchers. And the effect of actual sensor failure is just treated equally as that of attacks. Thereby, researchers just try to mitigate the consequences of both actual sensor failure and attacks, such as [92] [93].

I. Drawbacks of Existing Protection Methods

At present, research on sensor attacks on autonomous vehicle is still at an early stage. Methods of protection against sensor attacks mainly focus on a single type of sensor. Little attention has been paid to detection methods for the cases of multiple types of sensor being attacked. On the whole, there is no systematic theory or architecture for the detection of and defense against attacks. In addition, most existing protection methods focus on the detection of attacks. For an identified attack, there are no recovery methods for sensor data which are able to work in an intrusion-tolerant manner.

IV. Operating System Security

An autonomous driving system integrates multiple software modules, such as localization, perception, planning and control. These modules need to meet certain real-time requirements. Therefore, autonomous driving requires an operating system to manage these modules. The operating system mainly provides the functions of communication and resource allocation among the modules. Next, we discuss the security of the operating system. The sensors of an autonomous vehicle continuously generate data during their operation. The processing of data generated by each sensor imposes strong real-time requirements on the operating system. For example, for a camera with a frame rate of 60 FPS, the processing time for each frame is only about 16.66 milliseconds [94]. When the amount of data increases, allocating system resources becomes more difficult. For instance, when a large amount of LiDAR point cloud data is fed to the autonomous driving system, the subsequent process occupies considerable computing resources. It is likely that the camera data cannot be processed in time. Then, the recognition of a traffic light might be missed. Due to the strong connections among the modules in the autonomous driving system, effective communication and resource allocation among the modules become challenging.

A. Early Mobile Robot Operating Systems

Before autonomous driving, there were mainly three popular mobile robot operating systems.

1) Miro: Miro is an object-oriented robot middleware. It is used to improve the software development of mobile robots and provide interactions between the mobile robot and enterprise information processing systems. Technically, Miro implements an object-oriented design by adopting the common object request broker architecture (CORBA) standard [95]. This architecture provides interprocess and cross-platform interoperability of distributed robot systems. In addition, Miro also provides users with frequently used services, such as self-positioning, mapping, and path planning.

2) URBI: URBI is a universal robotic body interface based on a client/server architecture [96]. URBI does not provide a graphical programming interface. It just includes an independent language. This language is characterized by the ability to directly access and control the joints and sensors of mobile robots. Users can set up a computer cluster with URBI to run services cooperatively.

3) OpenRDK: OpenRDK is a modular management framework for designing distributed robot systems [97]. OpenRDK is implemented with C++. Developers only need to focus on implementing the functions of the modules: the interaction and information sharing among the modules are managed by the framework itself.

These three operating systems mainly provide a software component management framework for mobile robots. Since these operating systems lack software libraries and visual debugging tools, they are not suitable for autonomous vehicles. In fact, they are not used by any autonomous vehicles. Initially, the operating system of most autonomous vehicles was basically developed based on ROS.

B. ROS

ROS is a powerful and flexible robot programming framework. It is a distributed multiprocessing framework based on messaging. Many key components of autonomous driving are implemented on ROS, such as quaternion-based coordinate transformation [98], a robotic 3D mapping framework [99], and the positioning algorithm SLAM [100]. The message mechanism of ROS enables a modular design based on software functions. Each module is able to read and distribute messages, and the modules collaborate with each other by messages. The reasons why ROS is suitable for autonomous driving scenarios are as follows.

1) Comprehensive development support: ROS provides a uniform programming interface framework. Developers can focus on the development of specific algorithms and functional verification. Tasks can be uniformly implemented by ROS, such as configuration management, quick prototype building, overall operation, debugging, and verification of algorithms [101].

2) Flexible module configuration: ROS adopts a loose coupling strategy for the modules. The localization, perception, planning, and control modules of autonomous vehicles have relatively independent functions. These modules can be developed and debugged independently, and the exchange of data among them is carried out by the ROS publish/subscribe scheme [102]. A newly developed module can be easily integrated into an existing system based on ROS.

3) Abundant debugging tools: ROS possesses excellent compatibility with famous visualization tools, such as RViz [103], Gazebo [104], rpt [105], and Webviz [106]. As autonomous driving involves considerable processing of images and point cloud data, there are significant demands for visualization. For ROS, the compatibility with visualization tools greatly facilitate the development and testing of the underlying algorithms [107].

C. The Security of ROS

Attacks on sensors are external attacks that do not require access to the autonomous vehicle's operating system. Internal attacks involve hacking into the autonomous vehicle's operating system. The autonomous vehicle's operating systems implemented based on ROS have a common security issue: ROS does not provides authentication for messaging and node creation [108]. There are mainly two types of attack [4]: 1) a hijacked ROS node is able to continuously generate and distribute messages. This kind of malicious behavior might make the system run out of memory (OOM). Then, the autonomous vehicle's operating system would start to close ROS node processes. This would result in a crash of the operating system. 2) Messages sent by a hijacked topic or service of ROS may be tampered with or forged, thus leading to abnormal behavior of the operating system.

The first attack is rooted in the fact that ROS has no isolation mechanism, thus an ROS node is able to access system resources without any restriction. Linux container (LXC) [109] provides lightweight virtualization to isolate processes and system resources. It does not need an instruction interpretation mechanism or full virtualization. For autonomous driving, hosting an LXC saves 5% CPU overhead over running an application natively [110]. The source of the second attack resides in the fact that the messaging among nodes is not encrypted, thus attackers are able to obtain the message content readily [111].

SROS [112] is a set of security enhancements for ROS. There are transport layer security (TLS) support for communication within ROS, the use of x.509 certificate permitting chains of trust, definable namespace globbing for ROS node restrictions and permitted roles, covenant user-space tooling for the auto generation of node key pairs, audit ROS networks, and construct/train access control policies. In addition, AppArmor profile library templates are also provided. The templates allow users to harden or quarantine ROS based processes running on a Linux kernel. Zhang *et al*. [113] proposed an access control framework named AC4AV for autonomous driving vehicles. Different access control models are developed to protect in-vehicle data in real-time data and historical data.

Apollo 3.5 and later versions replace the original ROS middleware and use the Apollo Cyber RT middleware instead [114]. As ROS was not originally designed for autonomous driving, Cyber RT includes components designed to build autonomous driving modules and applications. The communication among components is conducted through the Cyber channel of Cyber RT, while the communication among modules is conducted by messages based on protocol buffers. Moreover, Cyber RT supports asynchronous computing, which greatly contributes to the optimization of thread usage and resource allocation. Unlike ROS, there is no master node in Cyber RT. The entire network topology of Cyber RT is divided into different domains. When a new node joins the network, it sends broadcast messages to other nodes in the domain with a real time publish subscribe (RTPS) protocol [115]. Each node which receives the broadcast messages sends its own information to the new node. The RTPS protocol handles the exchange of information based on the master node in ROS.

Xu *et al*. [116] deployed a data driven prediction architecture for autonomous driving on the Apollo platform. The architecture includes two main algorithms: 1) deployment of a semantic map on the Apollo platform for trajectory point prediction, 2) a trajectory generation method based on behavior prediction and its automatic parameter adjustment mechanism. The architecture enables rapid and efficient deployment of Apollo's prediction technologies across different regions.

D. Security Enhancement of ROS2

A data distribution service (DDS) [117] was first applied in the US Navy to handle the compatibility problem of a large number of software upgrades in the complex network environment of its ships [118]. At present, it has become a mandatory standard of the U.S. Department of Defense. DDS is widely used in the fields of defense, civil aviation, industrial control, etc. It has become a standard solution for data publish/subscribe in distributed real-time systems. An autonomous vehicle's operating system needs to establish a universal, high speed, and efficient DDS framework across

multiple cores, multiple CPUs, and multiple boards. This DDS framework adopts a publish/subscribe architecture. It focuses on data and provides a large number of quality of service (QoS) strategies. DDS is able to ensure a real-time, efficient, and flexible distribution of data and meets the needs of various distributed real-time communication applications. The security standard for DDS implements three-way handshakes which contains three messages: HandshakeRequest, HandshakeReply, and HandshakeFinal [119].

Participants are identified based on certificates using a public key infrastructure (PKI) and the Diffie–Hellman (DH) key exchange protocol. Assume there are two participants, A and B, which are identified by PKI. Firstly, participant A sends a HandshakeRequest message to initiate the DDS key exchange protocol. The HandshakeRequest message includes three parts: the certificate information of A, a DH public key, and a random code. Secondly, participant B sends a HandshakeReply message to respond to the HandshakeRequest message. The HandshakeReply message includes four parts: the certificate information of B, a DH public key, the random nonce given by A, and another random nonce. The entire HandshakeReply message is signed by B. Finally, A sends a HandshakeFinal message to acknowledge the reply from B. The HandshakeFinal message contains the above two random nonces. The entire HandshakeFinal message is signed by A. This process of three handshakes results in a secure channel between A and B. Then, the DDS protocol uses AES-GCM to encrypt and decrypt the data on this secure channel.

The DDS security specification defines five service plugin interfaces (SPIS) to increase security [120]:

- *Authentication Service Plugin*. This is central to the entire SPI architecture. It provides methods to verify the identity of an application or user that invokes operations on DDS. These methods include tools for performing mutual authentication between two participants and establishing a shared secret.
- *Access control service plugin*. This defines and enforces restrictions on the DDS-related capabilities of a domain participant. For example, it allows a user to restrict a particular participant to a specific DDS domain, or only allow the participant to read from or write to specific DDS topics, etc.
- *Cryptographic service plugin*. This handles all cryptography-related operations, including encryption, decryption, hashing, signature, etc. In addition, it includes methods of deriving keys from a shared key.
- *Logging service plugin*. This provides for the auditing of DDS security-related events. The generated log files are stored in the object storage service. Security audits on the database are performed by checking these log files (e.g., failure analyses).
- *Data tagging service plugin*. This tags specific DDS security-related actions performed by the users, providing the ability to add tags to data samples.

Unlike Apollo, Autoware [121] is currently developed based on ROS2 [122]. ROS2 has made significant improvements to the original ROS framework. It uses an advanced distributed architecture, rather than the original master–slave structure. ROS2 adopts DDS as its messaging model. The DDS security extensions are used to protect the data during transmission [123]. The adoption of DDS improves the reliability and real-time performance of multi-robot collaboration.

DDS is an industry standard implemented by many companies, such as RTI implementation Connext [124], eProsima implementation Fast DDS [125], and ADLINK implementation DDS [126]. There are many aspects to consider when choosing a DDS implementation, such as protocol legality and whether it is cross-platform. In order to prevent ROS2 from depending on a specific DDS program, ROS2 supports multiple implementations. Morita *et al*. [127] proposed a dynamic binding mechanism which is able to choose an appropriate DDS implementation.

Compared with ROS, ROS2 is enhanced in the following three aspects [128]:

- *Real-time*. DDS has a variety of transport configurations, such as deadline, fault-tolerance, and reliability. It brings real-time support to ROS2. For each deadline, both the data writer and data reader perform a data update at least once.
- *Continuity*. Although ROS has the concept of a data queue, it still has great limitations. For instance, subscribers cannot receive data before joining the network. But DDS can provide data history service for ROS. Even a newly added node can obtain all the previously released data.
- *Reliability*. Based on the DDS reliability configuration, users can choose the performance mode (BEST_EFFORT) or the stable mode (RELIABLE) according to their demands.

At present, the security of ROS2 is highly dependent on the security of DDS [119]. The implementation of ROS2 only employs the first three SPIS of DDS mentioned above.

- *A builtin authentication plugin (called "DDS: Auth: PKI-DH")*. This plugin uses a verified PKI. It requires each participant to have a public key, a private key, and an x.509 certificate. The x.509 certificate binds the participant's public key to a specific name. Each x.509 certificate must be signed by a specific certificate authority (CA). This CA is trusted by the plugin.
- *A builtin access control plugin (called "DDS: Access: Permission")*. This plugin also uses a PKI. It requires two signed XML documents per domain participant: a governance file and a permissions file. The former specifies how the domain should be secured. The latter contains the permissions of the domain participant and is bound to the name of the participant as defined by the above authentication plugin.
- *A builtin cryptographic plugin (called "DDS: Crypto:AES-GCM-GMAC")*. It provides authenticated encryption using advanced encryption standard (AES) in Galois counter mode (GCM), namely AES-GCM.

The main reason why ROS2 uses built-in plugins instead of other plugins is to allow all compatible DDS implementations

to be interoperable with ROS2. Thus, the security features of ROS2 are able to work with all vendors with minimal effort.

E. Drawbacks of ROS2

ROS2 lacks certain vital mechanisms. Here are two examples. 1) Secure over-the-air (OTA) update [129]: This establishes a connection between a background server of the vehicle manufacturer and an autonomous vehicle by WiFi. Update packages are downloaded from a server to update the local software of the vehicle. If the OTA is compromised by hackers, the security of autonomous vehicles will be affected. 2) Secure key exchange [130]: Current solutions for a communication channel for key exchange between remote listeners and talkers are not sufficiently secure. Thus, they are vulnerable to key interception attacks.

V. Control System Security

Various mechanical components and digital devices in autonomous vehicles are controlled by ECUs. The communication among different ECUs in a vehicle is conducted by a digital bus.

A. CAN

CAN is the main bus protocol of the in-car electronic network [131]. It has the advantages of stability and reliability, strong real-time performance, strong anti-jamming ability, and long transmission distance. A CAN bus adopts differential signal transmission. In general, its normal communication only needs two signal lines: CAN-H and CAN-L. The two possess opposed characteristics to avoid external electromagnetic interference and radiation [132]. In a CAN, a node can initiate communication to other nodes at any time. There is no master–slave relationship between the nodes. However, the right to use the bus is in accordance with node priorities. An autonomous vehicle often adds several telematics nodes in the CAN bus network [133]. As shown in Fig. 4, these nodes are connected to the CAN bus in order to facilitate remote control, remote upgrade, and other functions. Hackers can hack into the CAN bus network through the on-board diagnostics (OBD) port.

B. Vulnerabilities of CAN and Attack Methods

Currently, as the CAN bus has no authentication or access control, it is easily hijacked by hackers [134]. There have been many car network attacks against the CAN bus. In [135], the authors used system vulnerabilities to remotely control a Jeep's multimedia system. Then, they attacked the V850 controller and modified its firmware to obtain permission to remotely send commands to the CAN bus for the purpose of controlling the power system and braking system. This issue caused a recall of 1.4 million vehicles. In [136], the authors also attacked a Jeep's CAN bus and successfully controlled the steering, braking, acceleration, etc.

Generally speaking, it is difficult to get into the CAN bus itself. However, the entertainment system and the OBD-II port of the maintenance system are connected to the CAN bus. These connections expose possible attack paths to the CAN bus. Five popular attack paths are as follows.

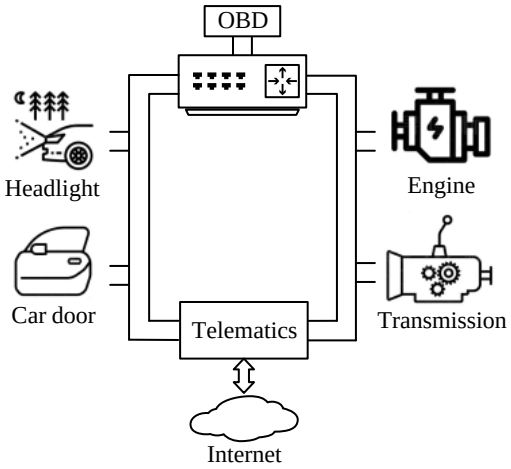

Fig. 4. CAN bus network.

1) OBD-II invasion: OBD-II improves OBD in terms of diagnostic functions and standardization. The OBD-II port is mainly used to access vehicle status. During vehicle maintenance, technicians use the detection software (e.g., Ford's NGS, Nissan's Consult II, Toyota's Diagnostic Tester) developed by vehicle vendors to manipulate the OBD-II port and examine the vehicle. Since the OBD-II port is connected to the CAN bus, hackers who have access to the detection software can easily intercept information on the CAN bus and control the vehicle [137].

2) Invasion of chargers for electric vehicles: Charging equipment is an essential component of an electric vehicle. The charging equipment also connects to the CAN bus. As the charging equipment of an electric vehicle communicates with an external charging pile, hackers have the opportunity to invade the CAN bus from the external charging pile [138].

3) CD player invasion: In general, a media player is connected to the CAN bus. Hackers can encode attack codes into a music CD. When the CD is played, the malicious codes invade the CAN bus from the CD player. Hence, the hackers are able to control the CAN bus [20].

4) TPMS invasion: TPMS stands for the tire pressure monitoring system. For the attack path, hackers inject attack codes into the TPMS. When the TPMS detects a specific value of tire pressure, the malicious codes are activated to attack the vehicle [139].

5) Bluetooth invasion: Autonomous vehicles support Bluetooth connections to other electronic devices (e.g., smartphones, personal digital assistants, and laptops). Malicious programs on smartphones are able to communicate with the CAN bus by the Bluetooth connection [140].

As the CAN bus lacks authentication, a CAN frame only indicates its destination. There is no information of the source of the message. As a result, malicious information can be regarded as valid information as long as the message format is correct. Based on this issue, the security protection methods for CAN bus fall into two categories: those based on encryption/authentication and those based on intrusion detection.

C. Protection Methods for CAN Bus

1) Methods based on encryption/authentication: These methods mainly conduct authentication for messages and ECUs or encrypt messages to ensure the security of the CAN bus. As the CAN bus lacks encryption schemes and the frame size is small, this kind of method often requires adding hardware to the ECUs or upgrading the existing firmware.

Groll *et al*. [141] employed a key distribution center in the vehicle network to divide the vehicle network into different areas. Different keys are assigned to these areas for communication. The communication among different areas are relayed by the central gateway. The feasibility of this proposal is evaluated by simulation on an embedded platform. The problem of clear text communication of the on-board ECU is addressed.

To prevent attackers from sniffing and tampering with the ECU codes, Yu *et al*. [142] used a Markov decision process to model the interaction between the attacker and the system and encrypted the storage system of the on-board ECU.

Murvay *et al*. [143] implemented a method for identifying the sources of the messages, based on an analysis of the frames on the bus. In particular, it is expected that a potential sender can be identified by measuring the voltage, filtering the signal, examining the mean square errors, and using convolutions.

Wang *et al*. [144] proposed a framework named Vecure to protect the CAN bus of vehicles. This framework uses the structure of a trust group to strengthen access control and prevent false messages from entering the CAN bus network. This framework adopts a message authentication scheme with offline computing capability, minimizing the message processing latency by pre-calculating a heavy-weighted cryptographic function. Experimental results show that Vecure's processing overhead for an individual message is only 50 μs.

Woo *et al*. [145] sent attack messages to the CAN bus network remotely through Bluetooth and OBD-II. For this attack, they presented a lightweight message encryption method based on the advanced encryption standard-128 (AES-128) algorithm. The performance of this proposal was compared with efficient protocol for secure broadcast (EPSB) algorithms [146] and ID table & message counter (IDT & C) [147] for different numbers of ECUs in terms of run-time, response time, and work load of the CAN bus.

2) Methods based on intrusion detection: Methods based on intrusion detection focus on establishing a detection model by analyzing the time series, frequency, and other characteristics of the messages. This kind of method introduces less overhead than using encryption and authentication. However, these methods require a more comprehensive understanding of a vehicle's CAN protocols. In addition, the false alarm rate of these methods is higher than that of the other kind.

Ning *et al*. [132] used local outlier factor (LOF) to identify attacks and detect intrusions in automotive networks. Data packets transmitted by different ECUs on the CAN bus produce distinct voltage waveforms. Each ECU corresponds to a unique voltage waveform. By identifying the voltage waveforms, this method is able to detect external malicious intrusion devices. A similar experimental study is presented in [148].

Song *et al*. [149] proposed a lightweight intrusion detection algorithm for a CAN bus based on an analysis of the time intervals of the CAN messages. This proposal is rooted in the periodicity of the CAN messages. The authors proved that the time interval is a vital element for detecting attacks and the proposal detected all message injection attacks in the experiments. However, their analysis of the intrusion data does not consider complex attack scenarios. Thus, the sensitivity of the detection is low. In addition, attacks aimed at aperiodic messages cannot be detected.

Taylor *et al*. [150] proposed an anomaly detection method based on the statistics of the traffic in the vehicle network. This method is able to detect injection attacks aimed at messages. However, it cannot detect attacks aimed at aperiodic messages.

Cho *et al*. [151] proposed a clock-based intrusion detection system which analyzes the clock offsets of the vehicle-mounted message timestamps to detect various attack scenarios. The system verifies a received message by extracting the clock offsets of the ECU and possesses a low false alarm rate.

Marchetti *et al*. [152] constructed multiple models of normal ID sequences of the collected messages based on the transition matrix group. Each model corresponds to one specific feature. This machine learning method detects injection attacks and possesses low computational overhead. However, the false alarm rate of this method is high.

Taylor *et al*. [153] proposed a learning model based on a long short term memory (LSTM) network to detect message sequences in the CAN bus. The learning model predicts the next data word from each sender on the bus. The advantage of the model is that there is no need to know the specific protocols of a vehicle. This model also possesses a high false alarm rate.

Kang *et al*. [154] studied an intrusion detection system using a deep neural network (DNN). The system employs probability-based feature vectors extracted from messages in a vehicle-mounted network to train the parameters of the DNN. For a given message, the DNN provides probability models for different types of message to distinguish between normal messages and attack messages.

Markovitz *et al*. [155] developed a greedy algorithm to split messages into different fields. Then, a semantically-aware anomaly detection system is built based on the field classification. The simulation yields a satisfactory false positive rate of 1%. However, the sensitivity of this model needs to be further evaluated with respect to actual attacks or modified simulated traffic.

D. CAN FD

CAN with flexible data-rate (CAN FD) was initially introduced as a specification [156] of BOSCH [157] in 2012. Then, it was formally presented in [158]. CAN FD is able to perform standard CAN communication. It shares the physical layer with the CAN as defined in the BOSCH CAN specification [159]. CAN FD can be considered as a protocol which provides efficient distributed real-time control with a high level of security. Safe data transfer, cogent error detection, signaling and self-checking are implemented in CAN FD node. Though

CAN FD is considered to be the next-generation in-vehicle network protocol, it has some security vulnerabilities suffered by CAN [160]. When a CAN data frame is broadcasted, the confidentiality and authentication are not guaranteed. CAN FD is also vulnerable to the above problem and suffers from eavesdropping and replay attacks.

Woo *et al*. [160] proposed a seven-phase security architecture for in-vehicle CAN FD. The design principles of the architecture are focused on four aspects: confidentiality, authentication, access control, and key management. Based on the analysis of attack models, the proposed architecture contains long-term symmetric key exchange, authenticated key exchange, and encryption/authentication of CAN FD data frames, etc.

Xie *et al*. [161] pointed out that CAN FD lacks a security authentication mechanism and is vulnerable to masquerade attacks. The authors developed a two-stage security enhancement for real-time parallel in-vehicle applications. The security is enhanced by adding message authentication codes (MACs) to messages. This is the first study towards security enhancement of CAN FD messages for real-time parallel in-vehicle application.

Xie *et al*. [162] proposed a security-aware obfuscated priority assignment approach for CAN FD messages. As this method is able to obtain tens of thousands of available obfuscated sequences efficiently, the scaling across effects of attacks on a group of autonomous vehicles can be effectively mitigated. This work is beneficial to automakers who suffer from the cascade effect and the recall problem due to automobile cyber attacks.

Xie *et al*. [163] developed a security enhancement method for independent in-vehicle CAN FD messages. The proposed method is able to dynamically adjust the message authentication code size of an independent message. Then, the payload of CAN FD message is maximized without compromising its hard real-time requirement.

Yu *et al*. [164] pointed out that unauthorized devices are able to access CAN FD by embedding external intruding devices to in-vehicle networks. The authors proposed an intrusion detection model based on verification of network topology. The above mentioned external intruding devices can be reliably detected with a random walk-based topology construction and subsequent verification.

Xie *et al*. [165] proposed an AUTOSAR-compliant system model which considers both time and security constraint. Here, AUTOSAR stands for AUTomotive Open System ARchitecture [166]. The model is defined as the basis for the design space exploration (DSE) method of CAN FD. The bandwidth overhead brought by the security constraints can be considered as a basis to evaluate different security enhancement mechanisms and configurations.

Xiao *et al*. [167] pointed out that a key security mechanism message authentication between ECUs for countering message spoofing and replay attack is crucial to the AUTOSAR-compliant system proposed in [165]. As the session key establishment with AUTOSAR compliance was not well addressed, the authors developed an AUTOSAR-compliant key management architecture. A baseline session key distribution protocol was designed based on the architecture. Moreover, a secret-sharing-based protocol was proposed for the purpose of improving communication efficiency.

Agrawal *et al*. [168] developed a security architecture for the communication between ECUs on different channels through gateway ECU (GECU). In specific, a group-based method was designed to secure communication between ECUs connected to the CAN FD network. Experimental results showed that using a fast authenticated encryption scheme AEGIS [169] was superior to using individual primitives for encryption and authentication in terms of real-time requirements.

E. Drawbacks of Existing Protection Methods

1) CAN: Most methods based on Encryption/Authentication require an update of the current CAN hardware. Moreover, these algorithms introduce extra computation into the CAN bus. This may affect the real-time performance of the CAN bus. Most existing methods based on intrusion detection can only be applied to a limited number of intrusion scenarios. Moreover, the actual performance of these methods is still unsatisfactory in terms of the false positive rate. In summary, both these types of protection methods contain complicated algorithms and introduce significant computational costs. Thus, the real-time requirements of a CAN bus are hardly met.

2) CAN FD: Though CAN FD is superior to CAN in terms of data payload size and bandwidth consumption, security is not well addressed for CAN FD. All attacks which are possible to CAN are also applicable for CAN FD [168]. With the increasing number of external intruding devices, the real-time performance of security enhancement built on topology construction/optimization is compromised. Moreover, popular security measures for CAN FD are based on encryption/authentication and intrusion detection methods, as well as for CAN. The design and implementation of these techniques are seriously confined by the real-time requirements for autonomous driving systems.

VI. V2X Communication Security

When an autonomous vehicle is on the road, it becomes part of the IoV. V2X is a catch-all term for the communication mechanisms of the IoV. As mentioned in Section I, these mechanisms usually include vehicle-to-vehicle, vehicle-to-infrastructure, vehicle-to-pedestrian, and vehicle-to-network. A vehicle can obtain a series of traffic information (e.g., real-time traffic status, pedestrians, status of surrounding vehicles) with V2X. Protecting the security of V2X communication is an important domain of autonomous driving. In this section, we discuss the potential security risks of V2X and corresponding solutions.

A. V2X Communication

The four kinds of communication in the V2X are shown in Fig. 5: V2V, V2I, V2P, and V2N. In V2V, the most common application scenarios are urban streets and highways, where

vehicles send data to each other for information sharing. This information includes the vehicle's speed, direction of motion, acceleration, braking, relative position, steering, etc. Data related to safety is shared among the neighboring vehicles. By predicting the driving behavior of other vehicles, a vehicle is able to take safety measures in advance. In V2I, vehicle-mounted devices communicate with the infrastructure point roadside units (RSUs). The RSUs obtain information about nearby vehicles and publish real-time information on Internet portals. In V2P, vehicles identify the behavior of nearby pedestrians with multiple sensors. When necessary, warnings can be issued with lights and the horn. It is expected that pedestrians will then become aware of the potential danger. In V2N, vehicle-mounted devices communicate with cloud servers to exchange information. The cloud stores and analyzes the uploaded data to provide various services to the vehicles, such as navigation, remote monitoring, emergency assistance, and in-car entertainment.

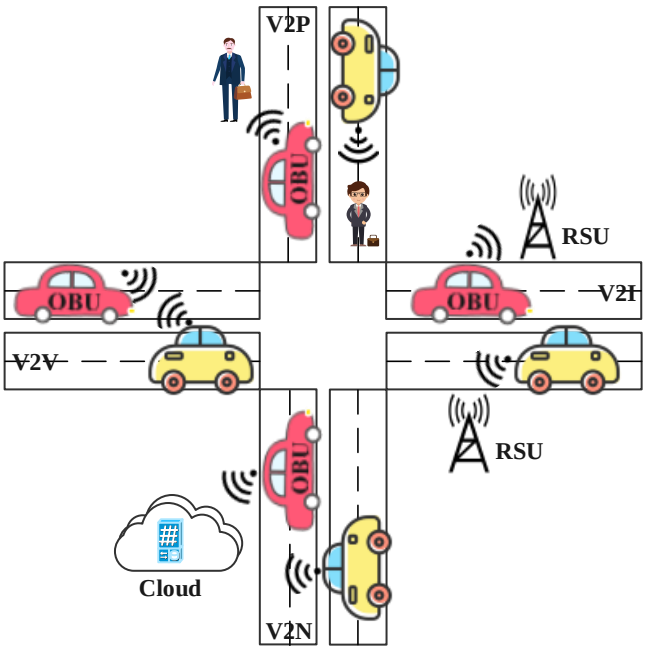

Fig. 5. V2X communication network.

B. V2X Communication Attacks and Solutions

Hasrouny *et al*. [170] presented a classification of attacks on V2X based on the compromised services. The attacks are classified into four groups: Authenticity/Identification, Availability, Data Integrity, and Confidentiality. Here, we conduct an in-depth study based on this classification and and review several notable publications. Representative studies of these four categories are summarized in Table I.

1) Authenticity/identification attacks and countermeasures:

- *Sybil attack*. In a vehicular ad hoc network (VANET), a vehicle joining the network becomes a wireless node. Since a node may join and leave a VANET freely, data is backed up among multiple nodes to enhance its availability to the network. A attacker may use a single malicious node to masquerade multiple identities, data being backed up in the same malicious node. Similarly, malicious messages can be propagated to other nodes by the same malicious node with multiple identities. For example, an attacker may propagate a fake traffic scene to several nodes. When another normal node in the network receives the fake traffic scene from those nodes, the normal node may modify its driving route. This may lead to a traffic accident [206]. Park *et al*. [171] proposed a detection method based on timestamp series. The method does not need a special vehicular PKI to authenticate a vehicle. A certificate can be issued by an RSU. The certificate signed by the RSU contains the current timestamp. In general, the likelihood of two vehicles passing through multiple RSUs simultaneously is vanishingly small. Thus, a Sybil attack can be detected by checking the similarity of the timestamp series. Li *et al*. [172] introduced a public key encryption model of pseudonym generation. This scheme allows a legitimate third party to obtain the real ID of a vehicle for identity authentication. The self-generated PKC-based pseudonyms are used as identifiers to protect privacy. Regional trusted authorities periodically propagate the current public key by RSUs. The current public key can be used for PKC-based generation of pseudonyms when a vehicle needs to update its current pseudonym or generate a new pseudonym. Yao *et al*. [173] proposed a method for detecting Sybil attacks based on vehicular voiceprints. Previous work based on RSSI calculates the relative position or absolute position based on the average value of the received signal strength indicator (RSSI). However, there is a common disadvantage in that illegal nodes which intentionally manipulate the transmission power during a Sybil attack cannot be identified. In [173], RSSI time series are used as vehicle-mounted speech to measure the similarity of the received series. Feng *et al*. [174] proposed an event-based reputation system (EBRS) to detect a Sybil attack on a VANET. In order to protect privacy, vehicles use pseudonyms instead of real identities to send messages in V2V communication and V2I communication. Each vehicle is issued a local certificate by a trusted RSU within its communication range. EBRS detects a Sybil attack by verifying the vehicle's local certificate. The EBRS establishes dynamic reputation and trusted value for each event message in VANET. If both the reputation value and the trusted value of a message are smaller than a given threshold, the message should not be propagated.
- *Key or certificate replication attack*. An attacker sniffers the network to obtain a certificate/key. The obtained credentials are then sent to an authentication server to declare its legal identify [207]. Azees *et al*. [208] proposed an efficient anonymous authentication scheme with conditional privacy-preserving (EAAP) to deal with key or certificate replication attacks on a VANET. The EEAP includes five main parts: 1) Registration and Key Generation, 2) Anonymous Certificate Generation, 3) Signature Generation, 4) Verification, and 5) Conditional

TABLE I
ATTACKS ON V2X.

Attack Classification	Attack	Attack Behavior
Authenticity/Identification	Sybil [171] [172] [173] [174]	Create multiple vehicles with the same identity
	Key or Certificate Replication [175] [176]	Steal certificates/keys
	GNSS Spoofing [177] [178]	Provide false location information
	Timing [179] [180]	Introduce transmission delay
Availability	DoS [181] [182] [183] [184]	Send numerous useless requests to compromise service availability
	DDoS [185]	Launch multiple DoS attacks from different nodes
	Spamming [186]	Send spam messages to consume bandwidth
	Flooding [187]	Broadcast false messages in the network
	Wormhole [188] [189]	Send packets over private channels
	Blackhole [190] [191] [192]	Discard relayed packets
	Malware [193]	Inject viruses into software by insiders
	Jamming [194] [195]	Transmit interference signals to communication channels
	Broadcast Tampering [196] [197]	Tamper with security messages in communication channels
Data Integrity	Masquerading [198]	Use a valid identity to hide
	Replay [199] [200]	Send previous messages repeatedly
	Illusion [201] [202]	Create fake traffic messages
	Message Alteration [203]	Modify, add, and discard the data packets
Confidentiality	Traffic Analysis [204]	Monitor the network and analyze packets to infer sensitive information
	Eavesdropping [205]	Obtain confidential data by unauthorized access

Tracking. The conditional tracking mechanism of the EAAP reveals the true identity of a malicious vehicle. Experimental results show that the EAAP has high computational efficiency during certificate and signature verification. Oulhaci *et al*. [176] proposed a distributed vehicle authentication architecture based on public keys. A trusted root authority is used to authorize a group of regional certification authorities. These certification authorities issue public key certificates to the vehicles. The root authority supervises the regional certification authorities. Each regional certification authority cooperates with its subordinate RSUs to sign public key certificates by threshold signature.

- *GNSS spoofing attack*. In a VANET, accurate and reliable location information is crucial to the operation of the whole network. An interference system designed by hackers generates false navigation signals which mislead the GNSS navigation of a vehicle. As the planning of autonomous vehicles is highly dependent on the sensor data, this attack is quite serious for an autonomous vehicle [209]. Curran *et al*. [177] proposed a method which uses uncalibrated low-cost IMUs to detect GNSS spoofing attacks. This method is based on a test of the coherency between the GNSS and the IMU. Both the norms of the acceleration vectors and the rotational rate vectors are extracted to conduct a comparison between the GNSS and the IMUs. However, a subsequent study [210] showed that ultrasonic pulses can stimulate certain microelectro mechanical systems (MEMS) sensors. This may cause IMUs to generate false measurements. Wang *et al*. [178] proposed a method based on edge computing to reconstruct unavailable and untrustworthy GPS signals. Firstly, the information of the GPS signals received by the edge nodes are cross-verified. Then, the GPS signals are reconstructed using the driving information (e.g., speed, steering angle) obtained from the CAN bus. The implementation of this method does not require the vehicles to carry any additional equipment (e.g., antenna, receiver).
- *Timing attack*. The timing attack is to delay the transmission of messages with high real-time requirements. As most messages with high real-time requirements are critical to the operation of a vehicle and the whole VANET, a malicious node in the network which introduces abnormal latency to certain messages is of great harm [211]. Chuang *et al*. [179] proposed a decentralized lightweight authentication framework called the trust-extended authentication mechanism (TEAM). This framework adopts a transferable trust relationship to improve the performance of the authentication process in a VANET. As it only uses an XOR operation and a hash function, the computational cost is acceptable. Arsalan *et al*. [180] proposed a protocol timing attack prevention (TAP) method based on a software defined network (SDN) [212], referred to as data networking (NDN) [213], to address the problem of the timing attack on a VANET. The protocol detects and mitigates the effect of an attacker vehicle in a software defined named data network (SDNDN). In an SDNDN with a VANET architecture, there is a centralized controller which manages and controls the entire network. When vehicle A sends a packet P to vehicle B, A's 2D Cartesian coordinates and vehicle ID are set to the packet. When vehicle B receives packet P, it records the packet arrival time (PAT). The time packet P arrived at vehicle A is called previous packet arrival time (PPAT). Vehicle B asks the centralized controller for the PPAT. In the detection phrase, vehicle B calculates the distance between the two vehicles by the coordinates. Given the signal's propagation speed, the theoretical propagation time between the two vehicles is obtained. Thus, whether a certain delay was introduced by vehicle A can be determined.

2) Availability attacks and countermeasures:

- *Denial of Service (DoS) attack*. A DoS attack aims to exhaust the resources of a VANET by sending a large number of useless requests. In this case, normal requests from valid users cannot get served. This type of attack

can be launched by malicious nodes inside or outside the network. When the network is filled with artificial malicious information, legitimate network nodes (e.g., on-board units, roadside units) are unable to work normally due to the scarcity of resources [214]. An enhanced version of the DoS attack is the distributed denial of service (DDoS) attack. An attacker can control a large number of victim nodes to perform many DoS attacks on a VANET. These victim nodes are called zombie nodes. There are two scenarios for the DDoS attack on a VANET [185]: 1) DDoS in V2V communication. Zombie nodes send message requests to a victim vehicle from different locations and time slots. For different types of nodes, the attacker can change the time slots and the content of the message requests. The attacker aims to overload the victim vehicle and bring down the network. As a result, the victim cannot access network resources. 2) DDoS in V2I communication. Attacks are launched from vehicles in different locations and the target is the RSUs. When the RSUs are overloaded, they are unable to respond to valid requests from normal nodes. Perrig *et al*. [181] proposed a timed efficient stream loss-tolerant authentication (TESLA) model. This proposal uses hash algorithm and single key encryption technologies to provide high quality authentication with low overhead. However, TESLA is vulnerable to memory-based DoS attacks. To address this problem, Studer *et al*. [182] proposed an effective authentication model for broadcast messages using symmetric cryptography and a delayed key. This model is called TESLA++, which is considered to be an improved version of TESLA. This model validates the broadcast messages and filters out malicious messages. The advantage of TESLA++ is the prevention of memory-based and computation-based DoS attacks. In addition, an authentication scheme named VAST is proposed to combine the advantages of fast authentication and non-repudiation of elliptic curve digital signature algorithm (ECDSA) with the advantage of TESLA++. The combination of ECDSA and TESLA++ is able to provide an effective authentication of messages in a VANET. Liu *et al*. [183] designed a puzzle-based co-authentication (PCA) scheme. This scheme includes two main parts: 1) Several hash puzzles are designed to limit the ability of attackers to forge certificates so as to carry out DoS attacks. 2) In identity authentication, trusted clusters are constructed among trusted vehicles. Authentication mechanisms based on trusted clusters reduce the computational overhead introduced by communication among legal vehicles. Thus, the authentication is accelerated. Jie *et al*. [184] proposed a mechanism to detect and filter malicious messages in a VANET by introducing port hopping [215] and a singular linear space [216]. This mechanism includes two parts: 1) A series of Anzahl formulas in singular linear space are used to construct a defense strategy in the form of matrices. 2) A simple port hopping method is proposed to deal with any vulnerabilities of the ports. For the messages in V2V and V2I, the attacker aims to detect UDP/TCP headers for vulnerable service ports. In this method, the port number of a service dynamically changes based on a function of time. By sharing cryptographic keys between server and user, the port number varies based on these keys.

- *Spamming attack*. Spamming attack is a type of denial-of-service attack (DoS) attack. In this type of attack, a large amount of spam is sent over the network to consume bandwidth, thereby increasing transmission delay on VANET [217]. Due to the lack of centralized management of the transmission medium, spamming control becomes considerably difficult. Malla *et al*. [186] proposed a redundancy elimination mechanism consisting of a rate decreasing algorithm and a state transition mechanism. This method is mainly divided into two steps: 1) Rate decreasing algorithm: Since there is a high chance of packet loss and connection loss in VANET, emergency warning messages (EVM) will be retransmitted at a certain rate, which might cause a broadcast storm or some extension of DoS attacks. Thus, the retransmission rate is reduced to half after a certain threshold time. If the nearest neighbor node detects that the source node does not decrease this rate, the intermediate node and the neighboring nodes will decide whether or not to block the source node based on the majority voting scheme. 2) State transition: All nodes must start from the initial abnormal state, starting transmitting EVM following rate decreasing algorithm reaches a state where the retransmission rate is 0 after a certain threshold time. Their experimental results show that this method can effectively control network traffic congestion, broadcast storm, and other malicious behaviors.
- *Flooding attack*. Flooding attack is also a type of DoS attack. The attacker broadcasts fake messages to the VANET through malicious nodes, which can consume a lot of resources and reduce the throughput of the network. In this case, the network stop service for a certain time period [218]. Faghihniya *et al*. [187] proposed a method, called the bus ad hoc on-demand distance vector (B-AODV) protocol, for detecting the route request (RREQ) flooding attack. In this method, balance index is obtained by combination of two statistical features: the mean number of RREQ and the dispersion number of RREQ. In addition, the balance index is used for detecting and preventing the RREQ flooding attack. The balance index is calculated when each node receives the RREQ packet. If the number of RREQs of the source node is greater than the balance index, the receiver node drops the RREQ. The experimental results obtained in this study show that B-AODV is capable of resisting flooding attacks and preventing the loss of network bandwidth.
- *Wormhole attack*. In VANET, a wormhole attack involves an attack in which the malicious nodes use the private channel already established in the network to transmit information that has been stolen from the network to another location in the network instead of transmitting it via a normal network connection. A malicious node can make any possible attack, such as packet dropping, data tampering, traffic analysis, etc., on the data passing

through the wormhole tunnel [219]. Safi *et al*. [188] used a packet leash and an authentication method called HEAP. The leash sets a maximum limit on the transmission distance of the packets in order to ensure that the communication distance between the sender and receiver of the data packet does not exceed the given maximum range. HEAP uses an improved hash message authentication code (HMAC)-based algorithm and two keys for authenticating each hop packet. The advantage of this approach is its low overhead. Ali *et al*. [189] used the public key cryptosystem RSA and symmetric key encryption technology to broadcast messages securely. In this method, the shared key is allocated using RSA and the packets are encrypted using the shared key. When a receiver receives the packet, it is decrypted using the shared key. By finding the identifier ID of the source node in the packet, the receiver can know the location of the sending node. By calculating the time at which the sending node sends the packet to the receiving node, the expected time at which the receiving node receives the packet can be determined. This method can effectively prevent a wormhole attack in VANET. However, if this method is applied to a VANET with a large number of nodes, a large amount of computation will be required, leading to substantial power consumption.

- *Blackhole attack*. The blackhole attack is a conventional attack against the availability of VANET. After receiving the routing request packet, the malicious nodes in the network will claim to be the nearest nodes with low latency to the destination node, and thus many nodes will choose them as the next-hop node for data packet forwarding. In the stage of data transmission, the malicious nodes usually directly discard the data packet without forwarding it. As a result, packet loss will occur in VANET [220]. Daeinabi *et al*. [190] proposed an algorithm called detection of malicious vehicles for detecting malicious vehicles that drop packets and isolate them from the normal vehicles. The algorithm introduces a distrust value to each vehicle, which is used for estimating the trustiness value of each vehicle when forwarding packets to them. Each vehicle is monitored by some of the more reliable verifier vehicles around it. If these verifier vehicles observe abnormal behavior in a vehicle, the distrust value of the vehicle will increase. When the distrust value of a vehicle is greater than a given threshold, its ID is reported to a third-party certification authority. The simulation results obtained from this study show that this algorithm can detect most malicious vehicles in VANET. Baiad *et al*. [191] proposed a cross-layer cooperative blackhole attack detection scheme that consists of three main layers of defense: 1) In the physical layer, each legitimate vehicle is assigned a signature key. A few trusted nodes are selected as the monitoring nodes of the physical layer. Based on the signature key and the maximum likelihood test, each of these monitoring nodes determines whether the detected node is a legitimate user or not. Since the physical layer might have interference because of noise or other physical signals and cause detection errors, further detection is required. 2) In the network layer, the watchdog monitoring technology is used for monitoring the messages transmitted from a source node to a destination node to further improve the verification reliability. Since the watchdog monitoring technology detects whether a legitimate conflict is malicious or not, it is necessary to further detection at the medium access control (MAC) layer. 3) In the MAC layer, the number of sent request to send (RTS) packets and the number of received clear to send (CTS) packets are calculated and compared to distinguish whether packet loss is caused by a conflict or an attack, in order to further reduce the false positive rate. Abdulkader *et al*. [192] proposed a routing protocol called lifetime improving ad hoc on-demand distance vector (LI-AODV) to deal with the blackhole attack in VANET. In this method, the RREQ message from the source node is sent to all of its neighboring nodes. These nodes receive the message and generate a route reply (RREP) return message. The authors choose the best path from the source node to the destination node by using a path rater. Based on the transmission of the RREQ and RREP messages among these nodes, they developed six rules to analyze the behavior of the nodes to determine the malicious node among them. When the malicious node is found, the path rater chooses the best alternative path in the network to continue the message transmission and deletes the detected malicious node in the network. The experimental results of this study show that the method has a higher true positive detection rate and a lower false-positive detection rate.
- *Malware attack*. In a malware attack, when on-board units (OBUs) and roadside units (RSUs) need patches or software updates, it is possible that malware, such as computer viruses, can disturb the operation of the network [221]. This type of attacker is usually a malicious insider rather than an outsider. Such an attack can be mitigated by using anti-malware or firewalls [193].
- *Jamming attack*. In a jamming attack, a moving vehicle is used as a node. The nodes communicate with each other by transmitting radio frequency (RF) signals. However, due to the low reliability of mobile computing and the high scalability of the system in a wireless environment, attackers can launch high-power interference signals to the communication channel, causing the node to reduce or even lose the ability to receive data packets [222]. Mokdad *et al*. [194] proposed a new algorithm, called DJAVAN, to detect interference attacks in VANET. This method calculates the packet delivery ratio (PDR) in the MAC layer. The PDR is calculated from the ratio of the number of packets that pass the cyclic redundancy check (CRC) to the number of packets received. The experimental results of this study showed that the PDR value decreases faster when the vehicle is closer to the jammer. Karagiannis *et al*. [195] proposed an unsupervised learning method to detect jamming attacks on vehicle communication. The previously developed methods cannot distinguish between intentional and unintentional interferences using various metrics such as signal-to-

interference-and-noise ratio (SINR), PDR, received signal strength, and interference (RSSI)). Therefore, the method developed by them utilizes a new metric, namely, the change in the relative speed between the jammer vehicle and the receiver vehicle and the parameters that can be obtained from the on-board wireless communication equipment on the receiving vehicle. In this study, the unique characteristics of each jamming attack were identified by unsupervised learning with clustering. Abderrahim *et al*. [223] proposed an analytical jamming model which is able to determine thresholds more accurately in threshold-based detection methods. However, threshold-based detection methods are not suitable for real-time applications. Therefore, a MAC-based real-time medium-access -control-based detection method is proposed. This method distinguishes competition conflict and jamming attack in VANET. Thus, the false alarm rate is reduced effectively.

- *Broadcast tampering attack*. In this type of attack, by injecting false security information into the network or tampering with the broadcast security messages, attackers force the legitimate vehicles to make choices that are not good for themselves, which might cause traffic accidents or increase the traffic flow on a certain road [224]. Wasef *et al*. [197] first described that PKI is a viable mechanism to protect VANET. Then, some limitations of PKI are also presented. For example, PKI does not meet security requirements such as location privacy, efficient authentication, and distributed and fair revocation. To address the above shortcomings, some complementary security mechanisms are introduced (e.g., appending a hash message authentication code to an outgoing signed message). Experimental results showed that the proposed mechanism alleviated DoS attacks in VANET. He *et al*. [196] proposed a lightweight and efficient broadcast authentication scheme, which mainly adopted a one-way hash chain and group key update technology. Instead of attaching an HMAC to the message, they appended a chain value of the one-way hash chain to the message. The receiver vehicle will verify the signature of the message only after the message has passed the pre-authentication based on the one-way hash chain; otherwise, it will refuse to verify the signature. Once abnormal behavior of a vehicle is detected, the trusted authority (TA) will immediately revoke its group key, and the abnormal vehicle cannot obtain the new group key. Experimental results show that their method can effectively alleviate broadcast tampering attacks and reduce resource consumption on VANET.

3) Data integrity attacks and countermeasures:

- *Masquerading attack*. In this type of attack, attackers use forged identities to gain informal access to the network. In this way, they can alter or discard data packets transmitted in VANET. An example of this type of attack is a malicious node disguising itself as an emergency vehicle to force other vehicles to slow down or stop [219] [225]. Malhi *et al*. [198] proposed a framework that uses genetic algorithms to detect and prevent masquerading attacks in VANET. In this framework, a vehicle calculates the fitness value using a fitness calculator and transmits this value to an RSU within its driving range. A comparator is used for comparing the fitness value calculated by the RSU to that calculated by the vehicle. If the comparator is able to match the two fitness values, a pseudonym generator assigns a pseudonym to the vehicle. Otherwise, the vehicle information is discarded, and the vehicle credentials are canceled.
- *Replay attack*. In this type of attack, malicious vehicles repeatedly send messages from a certain time period in the past to other vehicles, causing them to be cheated and thereby achieving the purpose of traffic jams. For example, a malicious vehicle saves messages about a traffic accident from a certain time period in the past and uses it to deceive other vehicles after the message expires [226]. Li *et al*. [199] evaluated the trustworthiness of traffic data and vehicle nodes and proposed an anti-resistant trust (ART) management scheme. In this scheme, the trustworthiness of the data and nodes are modeled and evaluated as two independent metrics, namely, data trust and node trust. Data trust is used for evaluating the reliability of the traffic data, whereas node trust is used for evaluating the degree of trust of the nodes in VANET. The data trust is evaluated based on the collected data and is sensed from multiple vehicles. The node trust is evaluated from two aspects, namely, the functional trust, which corresponds to the possibility of a node realizing its function, and recommendation trust, which corresponds to the recommendation credibility of a node with respect to other nodes. Experimental results from this study show that this scheme can accurately evaluate the trustworthiness of nodes and data in VANET and can deal with various types of malicious attacks. Alazzawi *et al*. [200] proposed a scheme to deal with the replay attack in VANET. The scheme consists of six stages: In phase A, the vehicle needs to send a "joining request" message to the nearest RSU, following which the RSU needs to establish a connection with a TA to verify the validity of the vehicle. In phase B, after the vehicle has obtained approval and a signature by the RSU, it will periodically broadcast beacon messages (including its position, speed, acceleration, etc.) to neighboring vehicles and nearby RSUs and insert a current timestamp into the message. In phase C, when the signature expires, the vehicle needs to send an "update signature" message to the same RSU to update the signature. Before the RSU processes the message, it first checks the validity of the timestamp. In phase D, the vehicle restarts broadcasting beacons and will continue even when it is within the range of another RSU. In phase E, the vehicle can update the expired signature by sending an "update signature" message to other RSUs. In Phase F, if an authenticated vehicle starts broadcasting false messages on VANET, the TA will continue to track the vehicle and revoke its permissions. Compared to the previous ID-based schemes [227] [228], the overall communication overhead of this scheme is

lower.

- *Illusion attack*. In an illusion attack, an attacker manages to fake sensor readings on their vehicle to create fake traffic messages and broadcasts them to the neighboring nodes to cause traffic jams [229]. Lo *et al*. [201] used a plausibility network checking module and a rule database to verify the credibility of the message, mainly by checking whether the timestamp, speed, and other element fields of the given message conform to the corresponding predefined ruleset of the rule database. If the verification fails, it will be discarded. Zacharias *et al*. [202] proposed a framework called the misbehavior detection system (MDS) to detect an illusion attack in VANET. The framework is based on two independent sensors (onboard camera sensor system, V2X communication sensor system) to measure the local traffic density and use it as evidence for specific traffic conditions. Multiple lines of evidence from two sensor systems are fused together using Dempster's rule to determine whether there is an illusion attack.
- *Message alteration attack*. In this type of attack, the attacker alters the data packets in the network by adding, deleting, or discarding the data, resulting in the data integrity being broken [230]. Zhu *et al*. [203] divided the network into multiple domains, in which an RSU is responsible for allocating group private keys to localize the management of vehicles. This scheme uses HMAC to replace the time-consuming certificate revocation list (CRL) checking. HMAC works by appending a MAC to the message to verify the source and integrity of the message. In this work, the authors have adopted collaborative message authentication between vehicles, which implies that only a small number of messages need to be validated per vehicle, greatly reducing the authentication burden. Their experimental results show that this scheme can satisfy the requirement of verifying hundreds of messages per second in VANET.

4) Confidentiality attacks and countermeasures:

- *Traffic analysis attack*. In this type of attack, the attacker analyzes the traffic messages in V2X communication, extracts and collects as much information as possible that is beneficial to them (e.g., location of the vehicle, driving path of the vehicle), and induces bad behavior in other vehicles, which violates the data confidentiality in VANET [231]. Cencioni *et al*. [204] proposed a communication protocol called vehicle-to-infrastructure privacy enforcement protocol (VIPER) to deal with traffic analysis attacks in VANET. In order to prevent the attacker from learning the identity of the message sender from the message field, VIPER uses universal re-encryption [232] to encrypt each message. In particular, the message sender uses the public key of an RSU and a secret encryption factor to encrypt the message, each relay vehicle re-encrypts the message using its own encryption factor, and the RSU decrypts the message using its own private key. As the relay vehicle re-encrypts each forwarded message with its own secret encryption factor, the message encoding constantly changes with each relay vehicle, making it impossible for an attacker to track the message. Experimental results have shown that VIPER performs well on two key performance metrics, namely, queue occupancy and message delivery time.
- *Eavesdropping attack*. Due to the broadcast nature of wireless communication of VANET, the communication among vehicles might be eavesdropped by illegal users. Eavesdropping attack is a common attack against confidentiality that is usually launched at the network layer. In this type of attack, the attacker obtains confidential data, such as the location data used to track a vehicle, for their own purposes [233]. Dai *et al*. [205] proposed a security framework based on indirect reciprocity. The framework assigns a scalar reputation to each vehicle node in VANET and this is used for estimating how dangerous each node is to the VANET. The sending node uses the consensus mechanism and an encryption algorithm in the blockchain technology to record the behavior of other vehicle nodes in order to prevent the reputation of VANET from being tampered with by malicious nodes. The authors also proposed reinforcement learning based on the action selection strategy for a node to select reliable relay vehicle nodes or to choose whether to receive messages from source nodes or not.

C. V2X Communication Simulators

The research of V2X communication security requires powerful experimental support. Since experiments in a real environment consume manpower and other material resources, excessive experiments may be a waste of time for immature autonomous driving technologies. Therefore, simulators play an important role in the research of V2X communication security. Generally speaking, two kinds of simulators are involved: network simulator and traffic simulator. Network simulators are used to test the performance of network protocols and applications, while traffic simulators are used to generate vehicle trajectories. Table II summarizes popular simulators for the research of V2X communication.

D. Drawbacks of Existing Countermeasures

Most existing countermeasures against V2X attacks require certain authentication schemes. As V2X related devices have limited computing resources and storage capacity, designing a secure and efficient authentication solution is quite challenging. Two key factors are as follows.

1) Lightweight: Most existing authentication protocols are based on elliptic curve or bilinear pairings. The protocols have high computational and communication overhead. For a large V2X communication network, lightweight solutions should be developed.

2) Mutual authentication: Most existing authentication protocols only conduct unilateral authentication. For instance, the receiver of a message can confirm the identity of the sender, while the sender cannot confirm whether the receiver is a legitimate user. Mutual authentication can guarantee a secure communication.

TABLE II
POPULAR SIMULATORS FOR NETWORKING AND COMMUNICATION IN AUTONOMOUS DRIVING.

Type	Simulator	Language	Platform	Open source	Reference
Network Simulator	NS-2 [234]	C++, OTCL	Cygwin, Linux, MacOS	✓	[171] [173] [179] [182] [183] [197] [203]
	NS-3 [235]	C++, Python	Cygwin, Linux, MacOS	✓	[194]
	OMNeT++ [236]	C++, NED	Windows, Linux, MacOS	✓	[192]
	GloMoSim [237]	C, Java	Windows, Linux	✓	[188] [199]
Traffic Simulator	SUMO [238]	C++, XML	Windows, Linux, MacOS	✓	[180]
	VanetMobiSim [239]	Java, XML	Windows, Linux	✓	[191]
	TraNS [240]	C++	Linux	✓	[204]
	MATLAB [241]	MATLAB	Windows, Linux, MacOS	✗	[176] [191] [198]

E. Blockchain-based Security Measures for Vehicular Networks

For the countermeasures elaborated in Section VI-B, there is another challenging factor: decentralization. Most existing authentication protocols need trusted third party organizations to complete key distribution and authentication functionalities. The security of these authentication protocols heavily relies on third party organizations. However, a centralized third party organization is likely to be compromised. The concept of decentralization should be introduced.

Since V2X communication is conducted in VANET, V2X communication security can be tackled from another perspective: a vehicular network. As a powerful mathematical package which is born with decentralization, blockchain has attracted much research attention [242] [243] [244].

Yang *et al*. [245] proposed a decentralized blockchain-based trust management system for vehicular networks. Messages received by a vehicle can be validated with neighboring vehicles by a Bayesian inference model. A block is constructed based on the validation results by RSUs. The trust blockchain is maintained by all the RSUs. This proposal is able to collect, calculate, and store trust values in vehicular networks.

Zhang *et al*. [246] developed an AI-enabled trust management system which is similar to the proposal in [245]. The AI package used in the system is deep learning algorithm. Trust is established based on message validation. The trustworthiness is managed by RSUs. This system is able to detect malicious vehicles efficiently.

Zheng *et al*. [247] proposed a blockchain-based secure computation offloading model for edge cloud offloading. To achieve consensus in vehicular networks, the authors developed a distributed hierarchical software-defined VANET (SDV) security architecture. The blockchain technique is used to conduct access control for the purpose of protecting the cloud from illegal offloading actions.

Li *et al*. [248] developed a fair and anonymous scheme for advertising in vehicular networks. The basis of the scheme is a blockchain-based ad dissemination framework. The fairness is achieved using the Merkel hash tree and smart contracts. The anonymity is ensured with zero-knowledge proof techniques.

Kudva *et al*. [249] proposed a method called proof of driving (PoD). It is used to randomize the selection of honest miners for generating the blocks efficiently for blockchain-based VANET applications. An efficient and fair selection of miners is achieved. Experimental results showed that high quality smaller consensus sets were obtained. These sets were effective on preventing malicious vehicle nodes from participating in consensus.

Ma *et al*. [250] developed a decentralized key management mechanism based on blockchain for VANET. The registration, update, and revocation of vehicle's public key are automatically conducted. Experimental results showed that the proposal was superior to traditional certificate-based PKI scheme in VANET.

Chen *et al*. [251] proposed a traceable and authenticated key negotiation scheme based on blockchain. The key availability is guaranteed with timeliness. The scheme can be used for data sharing and electric transactions among vehicles.

Kaur *et al*. [252] developed a blockchain-based authentication mechanism for vehicular fog infrastructure. A cross-datacenter authentication and key-exchange scheme based on blockchain and elliptic curve cryptography (ECC) was elaborated. The distributed ledger of blockchain maintains the network information while the highly secure ECC is used for mutual authentication between vehicles and RSUs.

For the above recent advances of blockchain-based security measures for vehicular networks, Table III shows their features and attacks which could be defensed.

TABLE III
FEATURES AND ATTACK DEFENSE OF BLOCKCHAIN-BASED MEASURES.

Scheme	Feature	Attack defense
[245]	trust management	message spoofing attack bad mouthing attack ballot stuffing attack data tempering
[246]	trust management	simple attack bad mouth attack zigzag attack
[247]	access control	identity masquerading
[248]	fairness & anonymity	free-riding attack double-claim attack repudiation attack forgeability attack DDoS attack & SPoF/C
[249]	miner selection	hash guessing attack transaction modification observe-act attack
[250]	key management	eavesdropping attack public key tampering attack DoS attack collusion attack
[251]	key negotiation	man-in-the-middle attack packet-dropping attack decryption failure attack
[252]	authentication	replay attack impersonation attack

VII. Discussion and Solution

A. Real-world Cases

Table IV presents a summary of the 2019 and 2020 autonomous vehicle disengagement reports released by the California Department of Motor Vehicles [253]. It provides a detailed description of the total mileage traveled by the test vehicles in the autonomous driving mode, the number of test vehicles, the total number of disengagements, and the miles per intervention (MPI). Among these indicators, the MPI is the primary measure. In 2019, the test vehicles of Waymo have the longest total mileage, with an average traveled distance of 13,219 miles and requiring a manual takeover. The test vehicles of AutoX have the least number of takeovers, with an average traveled distance of 10,684 miles and requiring a manual takeover. However, its total mileage is relatively short. Although Baidu only deployed four autonomous driving test vehicles, it ran a total of 108,300 miles in 2019 with only six manual takeovers. In 2020, the MPI of Waymo's autonomous vehicles rises to 29,945 miles which is the best performance among all autonomous driving companies. Cruise follows closely with its MPI increased from 12,221 miles in 2019 to 28,520 miles in 2020.

Besides the above summary of disengagement reports. We also studied some real-world cases which are related to the security of autonomous driving. Six representative cases of the four security dimensions elaborated previously are described as follows.

1) Sensor security: In May 2016, a Tesla Model S with autopilot enabled in it crashed into a towed truck while turning left on a highway in Florida, USA, causing the driver's death [254]. The self-driving car was equipped with the Mobileye EyeQ3 vision system mounted in the middle of the windshield, a millimeter-wave radar under the front bumper, and 12 ultrasonic sensors around the body. The camera view on the Tesla car was blocked when the white truck turned, and at the same time, coupled with the interference of strong ambient light, the camera could not recognize the vehicles on the ground. The installation position of the millimeter-wave radar was too low, and the height of the chassis of the truck was higher than the detection distance of the millimeter-wave radar, which led to the failure of radar perception. In the case of an ultrasonic radar, since its measurement distance is generally short, it is impossible to detect longitudinal obstacles when driving at high speed. In general, this accident shows that the combination of the Mobileye vision system with the perception of the millimeter-wave radar is insufficient for solving the situation in the accident.

In June 2020, a Tesla car with autopilot enabled in it collided with a white truck [255]. Eight cameras and 12 millimeter-wave radars were installed around the car's body. The cameras were used for object recognition, whereas the radars were mainly used for measuring and following the speed of the vehicle on its front and its recognition rate for complex types of static objects was not high. In the sensor fusion process, only when the camera recognizes the vehicle in its front, can it be called the speed measurement information of the radar. This is because the camera recognizes obstacles based on the illumination and the physical colors of the surroundings. In this accident, the color of the vehicle in front of the car and the surrounding environment was similar. In addition, interference of strong ambient light led to an erroneous judgment of the camera. It is believed that there were no obstacles in front of the car. Another reason could be the limitation of the training data used in the camera vision algorithm. The deep learning model might not have been able to classify the top of the truck box, which led to the failure of its perception.

2) Operating system security: In March 2018, in Arizona, USA, an unmanned Uber vehicle collided with a cyclist during a road test, causing the world's first unmanned driving accident in which a pedestrian died [256]. The unmanned vehicle was equipped with seven cameras, a 64-line LiDAR instrument, and multiple millimeter-wave radars. When the sensors detect pedestrians, the information is delivered to the central processing unit of the vehicle for processing in order to control the next move of the vehicle. According to the NTSB report, Uber found that the data collected by the camera, LiDAR, and radar on the unmanned vehicle were normal, and the LiDAR had detected the cyclist crossing the road 5.6 s before the accident. However, the classification of objects in the autonomous driving system was erroneous, and this led to the failure of the software to correctly predict the victim's category and movement trajectory. The automated emergency braking system is generally required to turn on 1.3 s before the collision, but Uber disabled this function to prevent erratic driving [257].

3) Control system security: In June 2015, two security experts, Charlie and Chris, used system vulnerabilities to remotely control a Chrysler Jeep car multimedia system to obtain permission for remotely sending commands to the CAN bus [258]. Without the user's knowledge, the driving speed of the car was reduced and ignition is turned off. Either the car engine suddenly braked or the brakes failed, causing 1.4 million vehicles to be returned to the factory for repair. This incident has exposed many security issues of the vehicle network, such as the use of the same cellular network for communicating with the device, lack of code signing, and no automatic update function, and these undoubtedly provide opportunities for hackers to attack vehicles. In addition, hackers can also use features such as the data flow entering the vehicle from the infrastructure to launch new attack channels against the vehicle.

4) V2X communication security: In November 2016, researchers from the Norwegian security service company Promon obtained the username and password of a Tesla APP account when they hacked into the user's mobile phone [259]. By logging into the Tesla Internet of Vehicles service platform, they could locate, track, unlock, and start the vehicle at any time, eventually leading to the vehicle being stolen. In January 2018, a hacker attacked the data server of the car-sharing service provider GoGet, using the company's server to access the company's fleet and download user information, resulting in the leakage of a large amount of private data of car owners [260].

The main reason for the above two incidents is that the

TABLE IV
AUTONOMOUS VEHICLE DISENGAGEMENT REPORTS.

Company	Test Mileage (mile)		# of Test Vehicles		# of Departures		Miles Per Intervention	
	2019	2020	2019	2020	2019	2020	2019	2020
Waymo	1,454,137.0	628,839.0	148	239	110	21	13,219.0	29,945.0
AutoX	32,054.0	40,734.0	8	8	3	2	10,684.0	20,367.0
Aurora	13,429.0	12,201.0	6	12	141	37	95.0	329.0
Cruise	831,040.0	770,049.0	228	137	68	27	12,221.0	28,520.0
Baidu	108,300.0	-	4	-	6	-	18,050.0	-
Apple	7,544.0	18,805.0	70	69	64	130	117.0	145.0
Nuro	68,762.0	55,370.0	33	20	34	11	2,022.0	5034.0

external network that the vehicle communicates with does not have a complete mechanism for encryption, authentication, and access control to prevent identity impersonation and information theft. Therefore, in the future design of the V2X communication network, in addition to reducing the network delay, it is necessary to strengthen the end-to-end encryption transmission, authentication, access control and abnormal traffic monitoring, and other security measures.

B. A Conceptual Vehicle Information Security Framework

In order to ensure the safe operation of autonomous vehicles while driving, a real-time monitoring system for autonomous vehicles should be designed for monitoring the environmental status, the status of the vehicle itself, the status of the autonomous driving hardware and software, and the status of the driver. From the judgment of various state changes, the corresponding prompts, warnings, and triggers of the takeover strategy should be carried out in order to ensure that the process of automatic driving is always controllable, safe, and reliable. After a problem has been identified, it is necessary to provide an online diagnostic system to help users quickly determine the problem of the automatic driving system and provide feasible solutions to help users restore the system to a usable and safe state as soon as possible.

In the present work, we have constructed a vehicle information security solution based on the multi-layer depth defense system. The main aim of the system is to "defend against external intrusions, prevent leakage of core applications and private data, and prevent threats of vehicle control". As shown in Fig. 6, the vehicle information security framework based on the multi-layer depth defense system is mainly divided into six layers: an external communication layer, an access gateway layer, a network defense layer, an in-car application layer, a system defense layer, and a control defense layer.

1) External communication layer: The complete PKI system issues certificates for the devices participating in the automatic driving system and provides the required key and certificate management services. Secure communication is provided among the devices of the autonomous driving system and between the cloud and the car terminal to ensure confidentiality, integrity, authenticity, and tamper-proof communication data. The security upgrade kit ensures the safety and reliability of the OTA communication.

2) Access gateway layer: The dedicated vehicle security gateway isolates and controls access between the vehicle network and the internet and vehicle subnetworks, and identifies instructions, detects and prevents abnormal network behavior and operation instructions of untrusted vehicles in order to ensure vehicle network security.

3) In-car application layer: Based on the chip hardware security, from operating system guidance to running applications, a credibility measurement is performed throughout the entire process to prevent the operating system, core applications, and data from being tampered with. The privacy system provides protection for the core intellectual property (IP) and important business value data.

4) Network defense layer: The deep packet inspection (DPI) [261] [262] system is deployed on the Internet of Vehicles platform to collect and analyze traffic and message content at key points in the vehicle network to detect abnormal network communication traffic and other behaviors and make audit records for subsequent security analysis.

5) System defense layer: By using security assessment, penetration testing, deployment of anti-distributed denial-of-service, WEB application firewall, and security log analysis tools on the cloud platform, the safe operation of the cloud platform is ensured. For mobile applications, the use of memory obfuscation technology, patented virtual machine encryption technology, high strong protection shell technology, etc., to ensure that the application will not be used by hackers for vehicle attacks.

6) Control security layer: By encrypting CAN bus communication, it is ensured that the messages transmitted by the CAN bus of autonomous vehicles are not hijacked by malicious users. By monitoring the vehicle-mounted ECU, the monitoring module can determine whether a certain ECU is invaded by a malicious user, i.e., illegally obtaining the right to use the CAN bus. Functional safety ensures that the functions of the various components of the vehicle control system can be operated and run smoothly.

The vehicle extracts information from the cloud and the external environment using the external communication layer and transmits this information to the access gateway layer. This layer uses a dedicated vehicle security gateway that is divided into two isolation areas, which isolate the vehicle network from the internet and vehicle sub-networks, and part of the information is transmitted to the black box and the security computer in the application layer of the car. For example, the data of various sensors are recorded, stored, and analyzed when the system requires the driver to control the car. Other equipment inside include the control of the accelerator and brake. The network defense layer uses the network isolation method to deploy the DPI system in the

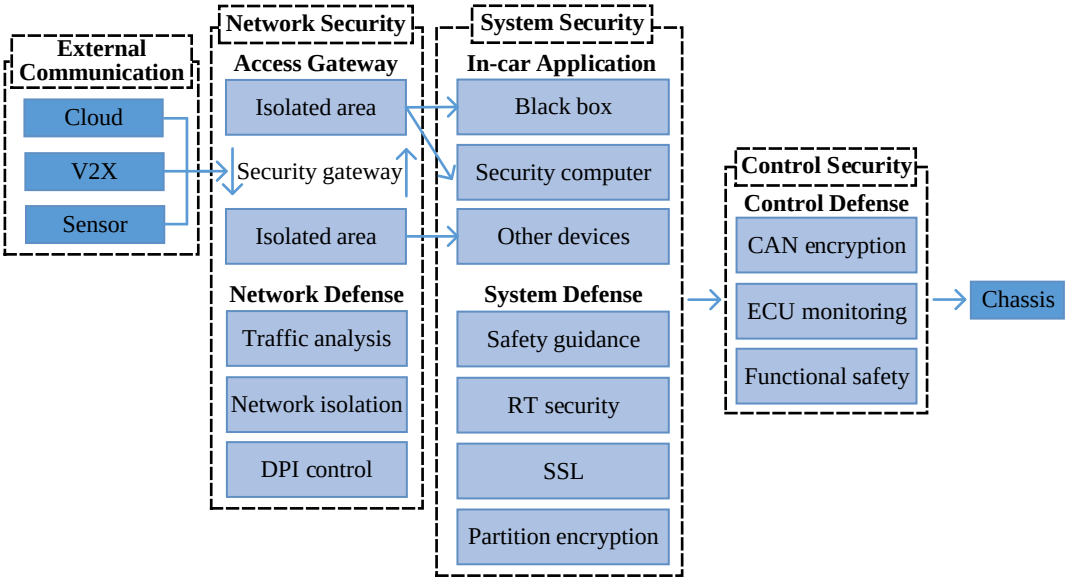

Fig. 6. Multi-layered defense architecture.

Internet of Vehicles for analyzing abnormal network behavior and for using the dynamic defense system to monitor and block network attacks in time. Finally, the system defense layer uses role-based trust (RT) management, secure sockets layer (SSL) certificates, partition encryption, and other operations to perform safety guidance of the vehicle. After passing through the system security layer, the vehicle needs to pass the executed instructions to control the security layer. The control security layer transmits control instructions to the actual components of the vehicle by encrypting CAN, ECU monitoring, and ensuring the functional safety of the control system.

For sensor security, we hold that actual sensor failure often shows its existence (e.g., obvious abnormal readings and sudden/dramatic changes of sensor data). On the contrary, the attacks against sensors are much sneakier. They tend to manipulate the sensor data and fool the high-level algorithms. Both actual sensor failure and attacks demand the autonomous driving system to operate in an error/intrusion-tolerant manner. We consider that sensors and sensor data can be covered in the "access gateway layer". This layer directly cooperates with the "external communication layer". There are some information which cannot be included in a PKI system of the "external communication layer", such as sunlight, rain, snow, fog, and shadows. Sensor data related to these phenomena can be cross-validated by multiple types of sensors with different data sources as described in Section III-F. In this case, both actual sensor failure and attacks may get compensated. Then, an autonomous driving system is expected to run in an error/intrusion-tolerant manner.

For operating system security, we hold that the major security drawback for the dominating operation system ROS2 in autonomous driving is it lacks protection for secure communication. We consider that the operating system security can be covered in the "external communication layer". In this layer, the security of operating system is firstly ensured from external communication and information which flows in and out ROS2. Then, the OTA communication is secured by a correct use of the security upgrade kit. Moreover, the operating system security can be covered in the "in-car application layer". In this layer, the chip hardware security ensures that from operating system guidance to running applications, a credibility measurement is performed throughout the entire process to prevent the operating system, core applications, and data from being tampered with. The "in-car application layer" also possesses black box, security computer, and other devices. These components interact with each other based on the technologies (e.g., RT security and SSL) provided by the module "System Defense" in Fig. 6. The above interactions are expected to achieve a secure operating environment for both the operating system and applications.

For control system security, we hold that the major drawbacks of existing protection methods for the control system are the significant computational cost and unsatisfactory real-time performance. We consider control system security can be covered in the "control security layer". In this layer, the measures we adopted are effective choices which are commonly accepted. These measures themselves do not show any improvement on computational cost or real-time performance. Nonetheless, the burdens on encryption and monitoring can be alleviated by partition encryption and RT security in the module "System Defense" and the three features in the module "Network Defense" illustrated in Fig. 6.

For V2X communication security, we hold that the major drawback of existing countermeasures for the V2X communication is the requirement of authentication. As is known to all, communication protocols based on authentication schemes often possess high computational overhead, as well as extra communication overhead. Thus, lightweight solutions are needed. Besides high overhead, unilateral authentication is another flaw in most existing authentication schemes used for V2X communication in autonomous driving. Though distributed

solutions (e.g., blockchain-based security measures described in Section VI-E) address the centralized problem, mutual authentication is still missing. Based on our investigation and literature review, current authentication/encryption schemes are unable to possess completeness and robustness with a lightweight design. We consider that the V2X communication security can be covered in the "external communication layer". In this layer, it is expected that the PKI system together with other supportive technologies are able to secure the V2X communication to some extent. Moreover, it can be covered in the "network defense layer". In this layer, traffic analysis conducted in vehicular network is expected to detect abnormal communication and other behaviors which might indicate a security issue.

VIII. Conclusion

Security is the primary requirement for autonomous driving. In this work, a retrospective and prospective study has been conducted in terms of four aspects: sensor security, operating system security, control system security, and V2X communication security. A detailed discussion of each attack path and the existing defense measures against these attack paths has been presented. The security problems of autonomous vehicles, caused by hackers intruding and tampering with data, belong to the category of information security, and thus a conceptual framework has been proposed in this work to build an efficient vehicle information security. However, if an autonomous vehicle is to be mass-produced, academia and industry still need to conduct additional research on the attack surface of autonomous driving modules. We hope that this paper will attract attention in the computer and automobile circles.

Acknowledgment

The authors would like to thank the anonymous reviewers whose comments and suggestions greatly helped us improve the quality and presentation of this paper.

References

[1] Philip Koopman and Michael Wagner. Autonomous vehicle safety: An interdisciplinary challenge. *IEEE Intelligent Transportation Systems Magazine*, 9(1):90–96, 2017.

[2] Kui Ren, Qian Wang, Cong Wang, Zhan Qin, and Xiaodong Lin. The security of autonomous driving: Threats, defenses, and future directions. *Proceedings of the IEEE*, 108(2):357–372, 2020.

[3] Qian Luo, Yurui Cao, Jiajia Liu, and Abderrahim Benslimane. Localization and navigation in autonomous driving: Threats and countermeasures. *IEEE Wireless Communications*, 26(4):38–45, 2019.

[4] Shaoshan Liu, Liyun Li, Jie Tang, Shuang Wu, and Jean-Luc Gaudiot. Creating autonomous vehicle systems. *Synthesis Lectures on Computer Science*, 6(1):i–186, 2017.

[5] Baidu Inc. Apollo. https://github.com/ApolloAuto/apollo. [Online; accessed 05-July-2021].

[6] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. ROS: An open-source robot operating system. In *2009 IEEE International Conference on Robotics and Automation (ICRA) Workshop on Open Source Software*, pages 1–6. IEEE, 2009.

[7] Michael Howard. Fending off future attacks by reducing attack surface. https://docs.microsoft.com/en-us/previous-versions/ms972812(v=msdn.10), February 2003. [Online; accessed 05-July-2021].

[8] Pratyusa K Manadhata and Jeannette M Wing. An attack surface metric. *IEEE Transactions on Software Engineering*, 37(3):371–386, 2011.

[9] Pratyusa Manadhata and Jeannette M Wing. Measuring a system's attack surface. Technical Report CMU-CS-04-102, Carnegie-Mellon Univ Pittsburgh PA School of Computer Science, January 2004.

[10] Pratyusa K Manadhata, Kymie M Tan, Roy A Maxion, and Jeannette M Wing. An approach to measuring a system's attack surface. Technical Report CMU-CS-07-146, Carnegie-Mellon Univ Pittsburgh PA School of Computer Science, August 2007.

[11] Pratyusa K Manadhata. Game theoretic approaches to attack surface shifting. In *Moving Target Defense II*, pages 1–13. Springer, 2013.

[12] Pratyusa Manadhata, Jeannette Wing, Mark Flynn, and Miles McQueen. Measuring the attack surfaces of two FTP daemons. In *2006 2nd ACM Workshop on Quality of Protection*, pages 3–10. ACM, 2006.

[13] Christopher Theisen, Nuthan Munaiah, Mahran Al-Zyoud, Jeffrey C Carver, Andrew Meneely, and Laurie Williams. Attack surface definitions: A systematic literature review. *Information and Software Technology*, 104:94–103, 2018.

[14] Syed Rizvi, RJ Orr, Austin Cox, Prithvee Ashokkumar, and Mohammad R Rizvi. Identifying the attack surface for IoT network. *Internet of Things*, 9:100162, 2020.

[15] Christopher Theisen, Kim Herzig, Brendan Murphy, and Laurie Williams. Risk-based attack surface approximation: How much data is enough? In *2017 IEEE/ACM 39th International Conference on Software Engineering: Software Engineering in Practice Track (ICSE-SEIP)*, pages 273–282. IEEE, 2017.

[16] Nuthan Munaiah and Andrew Meneely. Beyond the attack surface: Assessing security risk with random walks on call graphs. In *2016 ACM Workshop on Software PROtection*, pages 3–14. ACM, 2016.

[17] Christopher Theisen, Kim Herzig, Patrick Morrison, Brendan Murphy, and Laurie Williams. Approximating attack surfaces with stack traces. In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, volume 2, pages 199–208. IEEE, 2015.

[18] Carsten Maple, Matthew Bradbury, Anh Tuan Le, and Kevin Ghirardello. A connected and autonomous vehicle reference architecture for attack surface analysis. *Applied Sciences*, 9(23):5101, 2019.

[19] Martin Salfer and Claudia Eckert. Attack surface and vulnerability assessment of automotive electronic control units. In *2015 12th International Joint Conference on e-Business and Telecommunications (ICETE)*, volume 4, pages 317–326. IEEE, 2015.

[20] Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, Stefan Savage, Karl Koscher, Alexei Czeskis, Franziska Roesner, and Tadayoshi Kohno. Comprehensive experimental analyses of automotive attack surfaces. In *2011 20th USENIX Security Symposium (USENIX Security 11)*, volume 4, pages 447–462. USENIX, 2011.

[21] Intel Transportation Solutions Division. Research summary of the Intel automotive security research workshops. https://www.intel.com/content/dam/www/public/us/en/documents/product-briefs/automotive-security-research-workshops-summary.pdf, 2016. [Online; accessed 05-July-2021].

[22] Anupam Chattopadhyay, Kwok-Yan Lam, and Yaswanth Tavva. Autonomous vehicle: Security by design. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–15, 2020. doi:10.1109/TITS.2020.3000797 (Early Access).

[23] Derrick Dominic, Sumeet Chhawri, Ryan M Eustice, Di Ma, and André Weimerskirch. Risk assessment for cooperative automated driving. In *2016 2nd ACM Workshop on Cyber-physical Systems Security and Privacy*, pages 47–58. ACM, 2016.

[24] Charlie McCarthy, Kevin Harnett, and Art Carter. Characterization of potential security threats in modern automobiles: A composite modeling approach. Technical Report DOT HS 812 074, National Highway Traffic Safety Administration of United States, October 2014.

[25] Olaf Henniger, Ludovic Apvrille, Andreas Fuchs, Yves Roudier, Alastair Ruddle, and Benjamin Weyl. Security requirements for automotive on-board networks. In *2009 9th International Conference on Intelligent Transport Systems Telecommunications,(ITST)*, pages 641–646. IEEE, 2009.

[26] Jonathan Petit and Steven E Shladover. Potential cyber attacks on automated vehicles. *IEEE Transactions on Intelligent transportation systems*, 16(2):546–556, 2014.

[27] Kaile Xiao, Weisong Shi, Zhipeng Gao, Congcong Yao, and Xuesong Qiu. DAER: A resource pre-allocation algorithm of edge computing server by using blockchain in intelligent driving. *IEEE Internet of Things Journal*, 7(10):9291–9302, 2020.

[28] Yanlei Gu, Li-Ta Hsu, and Shunsuke Kamijo. Passive sensor integration for vehicle self-localization in urban traffic environment. *Sensors*, 15(12):30199–30220, 2015.

[29] Gorkem Kar, Hossen Mustafa, Yan Wang, Yingying Chen, Wenyuan Xu, Marco Gruteser, and Tam Vu. Detection of on-road vehicles emanating GPS interference. In *2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 621–632. ACM, 2014.

[30] Guillaume Bresson, Zayed Alsayed, Li Yu, and Sébastien Glaser. Simultaneous localization and mapping: A survey of current trends in autonomous driving. *IEEE Transactions on Intelligent Vehicles*, 2(3):194–220, 2017.

[31] Quanwen Zhu, Long Chen, Qingquan Li, Ming Li, Andreas Nüchter, and Jian Wang. 3D LiDAR point cloud based intersection recognition for autonomous driving. In *2012 IEEE Intelligent Vehicles Symposium*, pages 456–461. IEEE, 2012.

[32] Ashutosh Singandhupe and Hung La. A review of SLAM techniques and security in autonomous driving. In *2019 3rd IEEE International Conference on Robotic Computing (IRC)*, pages 602–607. IEEE, 2019.

[33] Wenyuan Xu, Chen Yan, Weibin Jia, Xiaoyu Ji, and Jianhao Liu. Analyzing and enhancing the security of ultrasonic sensors for autonomous vehicles. *IEEE Internet of Things Journal*, 5(6):5015–5029, 2018.

[34] Yutong Ye, Liming Fu, and Bijun Li. Object detection and tracking using multi-layer laser for autonomous urban driving. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 259–264. IEEE, 2016.

[35] Ming Yang, Shige Wang, Joshua Bakita, Thanh Vu, F Donelson Smith, James H Anderson, and Jan-Michael Frahm. Re-thinking CNN frameworks for time-sensitive autonomous-driving applications: Addressing an industrial challenge. In *2019 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 305–317. IEEE, 2019.

[36] Niall O'Mahony, Sean Campbell, Anderson Carvalho, Suman Harapanahalli, Gustavo Velasco Hernandez, Lenka Krpalkova, Daniel Riordan, and Joseph Walsh. Deep learning vs. traditional computer vision. In *2019 Science and Information Conference*, pages 128–144. Springer, 2019.

[37] Mark Campbell, Magnus Egerstedt, Jonathan P How, and Richard M Murray. Autonomous driving in urban environments: Approaches, lessons and challenges. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 368(1928):4649–4672, 2010.

[38] Marko Ilievski, Sean Sedwards, Ashish Gaurav, Aravind Balakrishnan, Atrisha Sarkar, Jaeyoung Lee, Frédéric Bouchard, Ryan De Iaco, and Krzysztof Czarnecki. Design space of behaviour planning for autonomous driving. *arXiv preprint arXiv:1908.07931*, 2019.

[39] Hong Cheng. *Autonomous intelligent vehicles: Theory, algorithms, and implementation*. Springer Science & Business Media, 2011.

[40] Christos Katrakazas, Mohammed Quddus, Wen-Hua Chen, and Lipika Deka. Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions. *Transportation Research Part C: Emerging Technologies*, 60:416–442, 2015.

[41] Mengxuan Zhang, Nan Li, Anouck Girard, and Ilya Kolmanovsky. A finite state machine based automated driving controller and its stochastic optimization. In *2017 Dynamic Systems and Control Conference*, pages 1–10. American Society of Mechanical Engineers (ASME), 2017.

[42] Sang-Hyeon Bae, Sung-Hyeon Joo, Jung-Won Pyo, Jae-Seong Yoon, Kwanghee Lee, and Tae-Yong Kuc. Finite state machine based vehicle system for autonomous driving in urban environments. In *2020 20th International Conference on Control, Automation and Systems (ICCAS)*, pages 1181–1186. IEEE, 2020.

[43] Jiajia Liu and Jianhao Liu. Intelligent and connected vehicles: Current situation, future directions, and challenges. *IEEE Communications Standards Magazine*, 2(3):59–65, 2018.

[44] Riccardo Marino, Stefano Scalzi, and Mariana Netto. Nested PID steering control for lane keeping in autonomous vehicles. *Control Engineering Practice*, 19(12):1459–1467, 2011.

[45] Wael Farag and Zakaria Saleh. Tuning of PID track followers for autonomous driving. In *2018 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT)*, pages 1–7. IEEE, 2018.

[46] Qingyang Zhang, Yifan Wang, Xingzhou Zhang, Liangkai Liu, Xiaopei Wu, Weisong Shi, and Hong Zhong. OpenVDAP: An open vehicular data analytics platform for CAVs. In *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*, pages 1310–1320. IEEE, 2018.

[47] Liangkai Liu, Sidi Lu, Ren Zhong, Baofu Wu, Yongtao Yao, Qingyang Zhang, and Weisong Shi. Computing systems for autonomous driving: State-of-the-art and challenges. *IEEE Internet of Things Journal*, 8(8):6469–6486, 2021.

[48] Zeinab El-Rewini, Karthikeyan Sadatsharan, Niroop Sugunaraj, Daisy Flora Selvaraj, Siby Jose Plathottam, and Prakash Ranganathan. Cybersecurity attacks in vehicular sensors. *IEEE Sensors Journal*, 20(22):13752–13767, 2020.

[49] Mario Hirz and Bernhard Walzel. Sensor and object recognition technologies for self-driving cars. *Computer-aided design and applications*, 15(4):501–508, 2018.

[50] Jonathan Petit, Bas Stottelaar, Michael Feiri, and Frank Kargl. Remote attacks on automated vehicles sensors: Experiments on camera and LiDAR. *Black Hat Europe*, 11(2015):995, 2015.

[51] Jiajun Lu, Hussein Sibai, Evan Fabry, and David Forsyth. Standard detectors aren't (currently) fooled by physical adversarial stop signs. *arXiv preprint arXiv:1710.03337*, 2017.

[52] Jindi Zhang, Yifan Zhang, Kejie Lu, Jianping Wang, Kui Wu, Xiaohua Jia, and Bin Liu. Detecting and identifying optical signal attacks on autonomous driving systems. *IEEE Internet of Things Journal*, 8(2):1140–1153, 2021.

[53] Yulong Cao, Ningfei Wang, Chaowei Xiao, Dawei Yang, Jin Fang, Ruigang Yang, Qi Alfred Chen, Mingyan Liu, and Bo Li. Invisible for both camera and LiDAR: Security of multi-sensor fusion based perception in autonomous driving under physical-world attacks. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 176–194. IEEE, 2021.

[54] Christopher DiPalma, Ningfei Wang, Takami Sato, and Qi Alfred Chen. Demo: Security of camera-based perception for autonomous driving under adversarial attack. In *2021 IEEE Security and Privacy Workshops (SPW)*, page 243. IEEE, 2021.

[55] Guodong Rong, Byung Hyun Shin, Hadi Tabatabaee, Qiang Lu, Steve Lemke, Mārtiņš Možeiko, Eric Boise, Geehoon Uhm, Mark Gerow, Shalin Mehta, Eugene Agafonov, Tae Hyung Kim, Eric Sterner, Keunhae Ushiroda, Michael Reyes, Dmitry Zelenkovsky, and Seonman Kim. LGSVL simulator: A high fidelity simulator for autonomous driving. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6. IEEE, 2020.

[56] Christos Kyrkou, Andreas Papachristodoulou, Andreas Kloukiniotis, Andreas Papandreou, Aris Lalos, Konstantinos Moustakas, and Theocharis Theocharides. Towards artificial-intelligence-based cybersecurity for robustifying automated driving systems against camera sensor attacks. In *2020 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pages 476–481. IEEE, 2020.

[57] Christian Vitale, Nikos Piperigkos, Christos Laoudias, Georgios Ellinas, Jordi Casademont, Pouria Sayyad Khodashenas, Andreas Kloukiniotis, Aris S Lalos, Konstantinos Moustakas, Pablo Barrientos Lobato, Javier Moreno Castillo, Petros Kapsalas, and Klaus-Peter Hofmann. The CARAMEL project: A secure architecture for connected and autonomous vehicles. In *2020 European Conference on Networks and Communications (EuCNC)*, pages 133–138. IEEE, 2020.

[58] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Conference on Robot Learning*, pages 1–16. PMLR, 2017.

[59] Mohamed Maher Atia, Allaa R Hilal, Clive Stellings, Eric Hartwell, Jason Toonstra, William B Miners, and Otman A Basir. A low-cost lane-determination system using GNSS/IMU fusion and HMM-based multistage map matching. *IEEE Transactions on Intelligent Transportation Systems*, 18(11):3027–3037, 2017.

[60] Tuan Li, Hongping Zhang, Zhouzheng Gao, Qijin Chen, and Xiaoji Niu. High-accuracy positioning in urban environments using single-frequency multi-GNSS RTK/MEMS-IMU integration. *Remote sensing*, 10(2):205–216, 2018.

[61] Rigas Themistoklis Ioannides, Thomas Pany, and Glen Gibbons. Known vulnerabilities of global navigation satellite systems, status, and potential mitigation techniques. *Proceedings of the IEEE*, 104(6):1174–1194, 2016.

[62] Vrizlynn L L Thing and Jiaxi Wu. Autonomous vehicle security: A taxonomy of attacks and defences. In *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. IEEE, 2016.

[63] Jaroslaw Magiera and Ryszard Katulski. Detection and mitigation of GPS spoofing based on antenna array processing. *Journal of applied research and technology*, 13(1):45–57, 2015.

[64] Shuai Han, Lei Chen, Weixiao Meng, and Cheng Li. Improve the security of GNSS receivers through spoofing mitigation. *IEEE Access*, 5:21057–21069, 2017.

[65] Sagar Dasgupta, Mizanur Rahman, Mhafuzul Islam, and Mashrur Chowdhury. Prediction-based GNSS spoofing attack detection for autonomous vehicles. *arXiv preprint arXiv:2010.11722*, 2020.

[66] Harald Schafer, Eder Santana, Andrew Haden, and Riccardo Biasini. A commute in data: The comma2k19 dataset. *arXiv preprint arXiv:1812.05752*, 2018.

[67] Roi Mit, Yoav Zangvil, and Dror Katalan. Analyzing Tesla's Level 2 autonomous driving system under different GNSS spoofing scenarios and implementing connected services for authentication and reliability of GNSS data. In *2020 33rd International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS+ 2020)*, pages 621–646. Institute of Navigation, 2020.

[68] Sagar Dasgupta, Tonmoy Ghosh, and Mizanur Rahman. A reinforcement learning approach for GNSS spoofing attack detection of autonomous vehicles. *arXiv preprint arXiv:2108.08628*, 2021.

[69] Vasili Ramanishka, Yi-Ting Chen, Teruhisa Misu, and Kate Saenko. Toward driving scene understanding: A dataset for learning driver behavior and causal reasoning. In *2018 IEEE Conference on Computer Vision and Pattern Recognition*, pages 7699–7707. IEEE, 2018.

[70] Ali Broumandan and Gérard Lachapelle. Spoofing detection using GNSS/INS/Odometer coupling for vehicular navigation. *Sensors*, 18(5):1305, 2018.

[71] Jiahui Song, Haitao Wu, Xuqiang Guo, Siyuan Li, Yingkui Gong, Yang Zhang, and Yaping Li. Credible navigation algorithm for GNSS attack detection using auxiliary sensor system. *Applied Sciences*, 11(14):6321, 2021.

[72] SH Jeong, CG Choi, JN Oh, PJ Yoon, BS Kim, M Kim, and KH Lee. Low cost design of parallel parking assist system based on an ultrasonic sensor. *International Journal of Automotive Technology*, 11(3):409–416, 2010.

[73] Chen Yan, Wenyuan Xu, and Jianhao Liu. Can you trust autonomous vehicles: Contactless attacks against sensors of self-driving vehicle. *Def Con*, 24(8):109, 2016.

[74] Bing Shun Lim, Sye Loong Keoh, and Vrizlynn L L Thing. Autonomous vehicle ultrasonic sensor vulnerability and impact assessment. In *2018 IEEE 4th World Forum on Internet of Things (WF-IoT)*, pages 231–236. IEEE, 2018.

[75] Jianzhi Lou, Qiben Yan, Qing Hui, and Huacheng Zeng. SoundFence: Securing ultrasonic sensors in vehicles using physical-layer defense. *arXiv preprint arXiv:2105.07574*, 2021.

[76] Karthik Ramasubramanian and Kishore Ramaiah. Moving from legacy 24 GHz to state-of-the-art 77-GHz radar. *ATZelektronik worldwide*, 13(3):46–49, 2018.

[77] Prateek Kapoor, Ankur Vora, and Kyoung-Don Kang. Detecting and mitigating spoofing attack against an automotive radar. In *2018 IEEE 88th Vehicular Technology Conference (VTC-Fall)*, pages 1–6. IEEE, 2018.

[78] Chao Zhou, Quanhua Liu, and Xinliang Chen. Parameter estimation and suppression for DRFM-based interrupted sampling repeater jammer. *IET Radar, Sonar & Navigation*, 12(1):56–63, 2017.

[79] Zhenyu Guan, Yongjiang Chen, Peng Lei, Dawei Li, and Ying Zhao. Application of hash function on FMCW based millimeter-wave radar against DRFM jamming. *IEEE Access*, 7:92285–92295, 2019.

[80] Zhi Sun, Sarankumar Balakrishnan, Lu Su, Arupjyoti Bhuyan, Pu Wang, and Chunming Qiao. Who is in control? Practical physical layer attack and defense for mmWave-based sensing in autonomous vehicles. *IEEE Transactions on Information Forensics and Security*, 16:3199–3214, 2021.

[81] Hocheol Shin, Dohyun Kim, Yujin Kwon, and Yongdae Kim. Illusion and dazzle: Adversarial optical channel exploits against LiDARs for automotive applications. In *2017 International Conference on Cryptographic Hardware and Embedded Systems*, pages 445–467. Springer, 2017.

[82] Yulong Cao, Chaowei Xiao, Benjamin Cyr, Yimeng Zhou, Won Park, Sara Rampazzi, Qi Alfred Chen, Kevin Fu, and Z Morley Mao. Adversarial sensor attack on LiDAR-based perception in autonomous driving. In *2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 2267–2281. ACM, 2019.

[83] Jiachen Sun, Yulong Cao, Qi Alfred Chen, and Z Morley Mao. Towards robust LiDAR-based perception in autonomous driving: General black-box adversarial sensor attack and countermeasures. In *2020 29th USENIX Security Symposium (USENIX Security 20)*, pages 877–894. USENIX, 2020.

[84] Raghu Changalvala and Hafiz Malik. LiDAR data integrity verification for autonomous vehicle using 3D data hiding. In *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1219–1225. IEEE, 2019.

[85] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012.

[86] Kaichen Yang, Tzungyu Tsai, Honggang Yu, Max Panoff, Tsung-Yi Ho, and Yier Jin. Robust roadside physical adversarial attack against deep learning in LiDAR perception modules. In *2021 ACM Asia Conference on Computer and Communications Security*, pages 349–362. ACM, 2021.

[87] Chengzeng You, Zhongyuan Hau, and Soteris Demetriou. Temporal consistency checks to detect LiDAR spoofing attacks on autonomous vehicle perception. *arXiv preprint arXiv:2106.07833*, 2021.

[88] Giedre Sabaliauskaite, Lin Shen Liew, and Jin Cui. Integrating autonomous vehicle safety and security analysis using STPA method and the six-step model. *International Journal on Advances in Security*, 11(1&2):160–169, 2018.

[89] Miguel Realpe, Boris X Vintimilla, and Ljubo Vlacic. A fault tolerant perception system for autonomous vehicles. In *2016 35th Chinese Control Conference (CCC)*, pages 6531–6536. IEEE, 2016.

[90] Nicolas Pous, Denis Gingras, and Dominique Gruyer. Intelligent vehicle embedded sensors fault detection and isolation using analytical redundancy and nonlinear transformations. *Journal of Control Science and Engineering*, 2017:1–10, 2017.

[91] Yeun-Sub Byun, Baek-Hyun Kim, and Rag-Gyo Jeong. Sensor fault detection and signal restoration in intelligent vehicles. *Sensors*, 19(15):3306, 2019.

[92] Md Toufiq Hasan Anik, Rachit Saini, Jean-Luc Danger, Sylvain Guilley, and Naghmeh Karimi. Failure and attack detection by digital sensors. In *2020 IEEE European Test Symposium (ETS)*, pages 1–2. IEEE, 2020.

[93] Alexandra Czarlinska and Deepa Kundur. Attack vs. failure detection in event-driven wireless visual sensor networks. In *2007 9th Workshop on Multimedia & Security*, pages 215–220. ACM, 2007.

[94] Jörn Migge, Josetxo Villanueva, Nicolas Navet, and Marc Boyer. Insights on the performance and configuration of AVB and TSN in automotive Ethernet networks. In *2018 9th European Congress Embedded Real-Time Software and Systems (ERTS 2018)*, pages 1–10. European Congress, 2018.

[95] Hans Utz, Stefan Sablatnog, Stefan Enderle, and Gerhard Kraetzschmar. Miro-middleware for mobile robot applications. *IEEE Transactions on Robotics and Automation*, 18(4):493–497, 2002.

[96] J-C Baillie. URBI: Towards a universal robotic low-level programming language. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 820–825. IEEE, 2005.

[97] Daniele Calisi, Andrea Censi, Luca Iocchi, and Daniele Nardi. OpenRDK: A modular framework for robotic software development. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1872–1877. IEEE, 2008.

[98] Asad Yousuf, Claire Cynthia Lehman, Mohamad A Mustafa, and Mir M Hayder. Introducing kinematics with robot operating system (ROS). In *2015 American Society for Engineering Education Annual Conference & Exposition*, pages 26–1024. ASEE, 2015.

[99] Janusz Będkowski, Michał Pełka, Karol Majek, Tresya Fitri, and Jacek Naruniec. Open source robotic 3D mapping framework with ROS—robot operating system, PCL—point cloud library and cloud compare. In *2015 International Conference on Electrical Engineering and Informatics (ICEEI)*, pages 644–649. IEEE, 2015.

[100] Zhen An, Lina Hao, Yu Liu, and Li Dai. Development of mobile robot SLAM based on ROS. *International Journal of Mechanical Engineering and Robotics Research*, 5(1):47–51, 2016.

[101] Open Robotics. ROS core components. https://www.ros.org/core-components. [Online; accessed 05-July-2021].

[102] Yukihiro Saito, Takuya Azumi, Shinpei Kato, and Nobuhiko Nishio. Priority and synchronization support for ROS. In *2016 IEEE 4th International Conference on Cyber-Physical Systems, Networks, and Applications (CPSNA)*, pages 77–82. IEEE, 2016.

[103] Hyeong Ryeol Kam, Sung-Ho Lee, Taejung Park, and Chang-Hun Kim. RViz: A toolkit for real domain data visualization. *Telecommunication Systems*, 60(2):337–345, 2015.

[104] Zandra B Rivera, Marco C De Simone, and Domenico Guida. Unmanned ground vehicle modelling in Gazebo/ROS-based environments. *Machines*, 7(2):42–63, 2019.

[105] Anil Mahtani, Luis Sanchez, Enrique Fernández, and Aaron Martinez. *Effective robotics programming with ROS*. Packt Publishing Ltd., 2016.

[106] Cruise LLC. Webviz. https://webviz.io/. [Online; accessed 05-July-2021].

[107] Morgan Quigley, Brian Gerkey, and William D Smart. *Programming robots with ROS: A practical introduction to the robot operating system*. O'Reilly Media, Inc., 2015.

[108] Se-Yeon Jeong, I-Ju Choi, Yeong-Jin Kim, Yong-Min Shin, Jeong-Hun Han, Goo-Hong Jung, and Kyoung-Gon Kim. A study on ROS vulnerabilities and countermeasure. In *the Companion of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*, pages 147–148. ACM, 2017.

[109] Canonical Ltd. Linuxcontainer. https://linuxcontainers.org. [Online; accessed 05-July-2021].

[110] Shaoshan Liu, Jie Tang, Chao Wang, Quan Wang, and Jean-Luc Gaudiot. Implementing a cloud platform for autonomous driving. *arXiv preprint arXiv:1704.02696*, 2017.

[111] Jarrod McClean, Christopher Stull, Charles Farrar, and David Mascarenas. A preliminary cyber-physical security assessment of the robot operating system (ROS). In *2013 SPIE Unmanned Systems Technology XV*, volume 8741, page 874110. International Society for Optics and Photonics (SPIE), 2013.

[112] Ruffin White, Henrik I Christensen, and Morgan Quigley. SROS: Securing ROS over the wire, in the graph, and through the kernel. *arXiv preprint arXiv:1611.07060*, 2016.

[113] Qingyang Zhang, Hong Zhong, Jie Cui, Lingmei Ren, and Weisong Shi. AC4AV: A flexible and dynamic access control framework for connected and autonomous vehicles. *IEEE Internet of Things Journal*, 8(3):1946–1958, 2020.

[114] Tobias Kessler, Julian Bernhard, Martin Buechel, Klemens Esterle, Patrick Hart, Daniel Malovetz, Michael Truong Le, Frederik Diehl, Thomas Brunner, and Alois Knoll. Bridging the gap between open source software and vehicle hardware for autonomous driving. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 1612–1619. IEEE, 2019.

[115] Daniel Heß, Stephan Lapoehn, Fabian Utesch, Martin Fischer, Julian Schindler, Tobias Hesse, and Frank Köster. Contributions of the EU projects UnCoVerCPS and enable-S3 to highly automated driving in conflict situations. In *2019 Annual Conference of American Association of Electrodiagnostic Technologists*, pages 1–25. AAET, 2019.

[116] Kecheng Xu, Xiangquan Xiao, Jinghao Miao, and Qi Luo. Data driven prediction architecture for autonomous driving and its application on Apollo platform. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 175–181. IEEE, 2020.

[117] Gerardo Pardo-Castellote. OMG data-distribution service: Architectural overview. In *2003 23rd International Conference on Distributed Computing Systems Workshops*, pages 200–206. IEEE, 2003.

[118] Cihat Eryigit and Sima Uyar. Integrating agents into data-centric naval combat management systems. In *2008 23rd International Symposium on Computer and Information Sciences*, pages 1–4. IEEE, 2008.

[119] Jongkil Kim, Jonathon M Smereka, Calvin Cheung, Surya Nepal, and Marthie Grobler. Security and performance considerations in ROS2: A balancing act. *arXiv preprint arXiv:1809.09566*, 2018.

[120] Object Management Group. DDS security. https://www.omg.org/spec/DDS-SECURITY/1.1, July 2018. [Online; accessed 05-July-2021].

[121] Autoware Foundation. Autoware. https://github.com/Autoware-AI/autoware.ai/. [Online; accessed 05-July-2021].

[122] Michael Reke, Daniel Peter, Joschua Schulte-Tigges, Stefan Schiffer, Alexander Ferrein, Thomas Walter, and Dominik Matheis. A self-driving car architecture in ROS2. In *2020 International SAUPEC/RobMech/PRASA Conference*, pages 1–6. IEEE, 2020.

[123] Vincenzo DiLuoffo, William R Michalson, and Berk Sunar. Robot Operating System 2: The need for a holistic security approach to robotic architectures. *International Journal of Advanced Robotic Systems*, 15(3):1–15, 2018.

[124] Real-Time Innovations. Software system integration with Connext DDS Professional. https://www.rti.com/products/connext-dds-professional. [Online; accessed 05-July-2021].

[125] eProsima. eProsima Fast DDS. https://www.eprosima.com/index.php/products-all/eprosima-fast-dds. [Online; accessed 05-July-2021].

[126] ADLINK Technology Inc. Data Distribution Service. https://www.adlinktech.com/en/data-distribution-service.aspx. [Online; accessed 05-July-2021].

[127] Ren Morita and Katsuya Matsubara. Dynamic binding a proper DDS implementation for optimizing inter-node communication in ROS2. In *2018 IEEE 24th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, pages 246–247. IEEE, 2018.

[128] Yuya Maruyama, Shinpei Kato, and Takuya Azumi. Exploring the performance of ROS2. In *2016 13th International Conference on Embedded Software*, pages 1–10. ACM, 2016.

[129] Roland Herberth, Sidney Körper, Tim Stiesch, Frank Gauterin, and Oliver Bringmann. Automated scheduling for optimal parallelization to reduce the duration of vehicle software updates. *IEEE Transactions on Vehicular Technology*, 68(3):2921–2933, 2019.

[130] John Lawrence. ROS2 prevalance and security. Technical Report CSEC 793, Rochester Institute of Technology, May 2020.

[131] Steve Corrigan. Introduction to the controller area network (CAN). Application Report SLOA101, Texas Instruments, August 2002.

[132] Jing Ning, Jiadai Wang, Jiajia Liu, and Nei Kato. Attacker identification and intrusion detection for in-vehicle networks. *IEEE Communications Letters*, 23(11):1927–1930, 2019.

[133] Bo Wang, Smruti Panigrahi, Mayur Narsude, and Amit Mohanty. Driver identification using vehicle telematics data. Technical Report 2017-01-1372, SAE Technical Paper, January 2017.

[134] Clinton Young, Joseph Zambreno, Habeeb Olufowobi, and Gedare Bloom. Survey of automotive controller area network intrusion detection systems. *IEEE Design & Test*, 36(6):48–55, 2019.

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

[136] Andy Greenberg. The Jeep hackers are back to prove car hacking can get much worse. https://www.wired.com/2016/08/jeep-hackers-return-high-speed-steering-acceleration-hacks/, August 2016. [Online; accessed 05-July-2021].

[137] Karl Koscher, Alexei Czeskis, Franziska Roesner, Shwetak Patel, Tadayoshi Kohno, Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, and Stefan Savage. Experimental security analysis of a modern automobile. In *2010 IEEE Symposium on Security and Privacy*, pages 447–462. IEEE, 2010.

[138] Sam Abbott-McCune and Lisa A Shay. Intrusion prevention system of automotive network CAN bus. In *2016 IEEE International Carnahan Conference on Security Technology (ICCST)*, pages 1–8. IEEE, 2016.

[139] Mahammad A Hannan, Aini Hussain, and Salina A Samad. System interface for an integrated intelligent safety system (ISS) for vehicle applications. *Sensors*, 10(2):1141–1153, 2010.

[140] Shaoshan Liu, Liangkai Liu, Jie Tang, Bo Yu, Yifan Wang, and Weisong Shi. Edge computing for autonomous driving: Opportunities and challenges. *Proceedings of the IEEE*, 107(8):1697–1716, 2019.

[141] Andre Groll and Christoph Ruland. Secure and authentic communication on existing in-vehicle networks. In *2009 IEEE Intelligent Vehicles Symposium*, pages 1093–1097. IEEE, 2009.

[142] Lu Yu, Juan Deng, Richard R Brooks, and Seok Bae Yun. Automobile ECU design to avoid data tampering. In *2015 10th Annual Cyber and Information Security Research Conference (CISR)*, pages 1–4. ACM, 2015.

[143] Pal-Stefan Murvay and Bogdan Groza. Source identification using signal characteristics in controller area networks. *IEEE Signal Processing Letters*, 21(4):395–399, 2014.

[144] Qiyan Wang and Sanjay Sawhney. VeCure: A practical security framework to protect the CAN bus of vehicles. In *2014 International Conference on the Internet of Things (IOT)*, pages 13–18. IEEE, 2014.

[145] Samuel Woo, Hyo Jin Jo, and Dong Hoon Lee. A practical wireless attack on the connected car and security protocol for in-vehicle CAN. *IEEE Transactions on intelligent transportation systems*, 16(2):993–1006, 2014.

[146] Bogdan Groza and Stefan Murvay. Efficient protocols for secure broadcast in controller area networks. *IEEE Transactions on Industrial Informatics*, 9(4):2034–2042, 2013.

[147] Chung-Wei Lin and Alberto Sangiovanni-Vincentelli. Cyber-security for the controller area network (CAN) communication protocol. In *2012 International Conference on Cyber Security*, pages 1–7. IEEE, 2012.

[148] Jing Ning and Jiajia Liu. An experimental study towards attacker identification in automotive networks. In *2019 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6. IEEE, 2019.

[149] Hyun Min Song, Ha Rang Kim, and Huy Kang Kim. Intrusion detection system based on the analysis of time intervals of CAN messages for in-vehicle network. In *2016 International Conference on Information Networking (ICOIN)*, pages 63–68. IEEE, 2016.

[150] Adrian Taylor, Nathalie Japkowicz, and Sylvain Leblanc. Frequency-based anomaly detection for the automotive CAN bus. In *2015 World Congress on Industrial Control Systems Security (WCICSS)*, pages 45–49. IEEE, 2015.

[151] Kyong-Tak Cho and Kang G Shin. Fingerprinting electronic control units for vehicle intrusion detection. In *2016 25th USENIX Security Symposium (USENIX Security 16)*, pages 911–927. USENIX, 2016.

[152] Mirco Marchetti and Dario Stabili. Anomaly detection of CAN bus messages through analysis of ID sequences. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 1577–1583. IEEE, 2017.

[153] Adrian Taylor, Sylvain Leblanc, and Nathalie Japkowicz. Anomaly detection in automobile control network data with long short-term memory networks. In *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 130–139. IEEE, 2016.

[154] Min-Joo Kang and Je-Won Kang. Intrusion detection system using deep neural network for in-vehicle network security. *PloS one*, 11(6):e0155781, 2016.

[155] Moti Markovitz and Avishai Wool. Field classification, modeling and anomaly detection in unknown CAN bus networks. *Vehicular Communications*, 9:43–52, 2017.

[156] Bosch Global. CAN with Flexible Data-rate Specification Version 1.0. Technical report, Robert Bosch GmbH, April 2012.

[157] Bosch Global. https://www.bosch.com.

[158] Florian Hartwich. CAN with Flexible Data-rate. In *2012 13th international CAN Conference (iCC)*, pages 1–9. CAN in Automation, 2012.

[159] Bosch Global. CAN Specification Version 2.0. Technical report, Robert Bosch GmbH, September 1991.

[160] Samuel Woo, Hyo Jin Jo, In Seok Kim, and Dong Hoon Lee. A practical security architecture for in-vehicle CAN-FD. *IEEE Transactions on Intelligent Transportation Systems*, 17(8):2248–2261, 2016.

[161] Guoqi Xie, Laurence T Yang, Wei Wu, Keyu Zeng, Xiangzhen Xiao, and Renfa Li. Security enhancement for real-time parallel in-vehicle applications by CAN FD message authentication. *IEEE Transactions on Intelligent Transportation Systems*, 22(8):5038–5049, 2021.

[162] Guoqi Xie, Renfa Li, and Shiyan Hu. Security-aware obfuscated priority assignment for CAN FD messages in real-time parallel automotive applications. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 39(12):4413–4425, 2020.

[163] Guoqi Xie, Laurence T Yang, Yao Liu, Haibo Luo, Xin Peng, and Renfa Li. Security enhancement for real-time independent in-vehicle CAN-FD messages in vehicular networks. *IEEE Transactions on Vehicular Technology*, 70(6):5244–5253, 2021.

[164] Tianqi Yu and Xianbin Wang. Topology verification enabled intrusion detection for in-vehicle CAN-FD networks. *IEEE Communications Letters*, 24(1):227–230, 2019.

[165] Yong Xie, Gang Zeng, Ryo Kurachi, Hiroaki Takada, and Guoqi Xie. Security/timing-aware design space exploration of CAN FD for automotive cyber-physical systems. *IEEE Transactions on Industrial Informatics*, 15(2):1094–1104, 2018.

[166] Simon Fürst and Markus Bechter. AUTOSAR for connected and autonomous vehicles: The AUTOSAR adaptive platform. In *2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshop (DSN-W)*, pages 215–217. IEEE, 2016.

[167] Yang Xiao, Shanghao Shi, Ning Zhang, Wenjing Lou, and Y Thomas Hou. Session key distribution made practical for CAN and CAN-FD message authentication. In *2020 20th Annual Computer Security Applications Conference*, pages 681–693. ACM, 2020.

[168] Megha Agrawal, Tianxiang Huang, Jianying Zhou, and Donghoon Chang. CAN-FD-sec: Improving security of CAN-FD protocol. In *Security and Safety Interplay of Intelligent Software Systems*, pages 77–93. Springer, 2018.

[169] Hongjun Wu and Bart Preneel. AEGIS: A fast authenticated encryption algorithm. In *2013 International Conference on Selected Areas in Cryptography (SAC)*, pages 185–201. Springer, 2014.

[170] Hamssa Hasrouny, Abed Ellatif Samhat, Carole Bassil, and Anis Laouiti. VANET security challenges and solutions: A survey. *Vehicular Communications*, 7:7–20, 2017.

[171] Soyoung Park, Baber Aslam, Damla Turgut, and Cliff C Zou. Defense against sybil attack in the initial deployment stage of vehicular ad hoc network based on roadside unit support. *Security and Communication Networks*, 6(4):523–538, 2013.

[172] Jie Li, Huang Lu, and Mohsen Guizani. ACPN: A novel authentication framework with conditional privacy-preservation and non-repudiation for VANETs. *IEEE Transactions on Parallel and Distributed Systems*, 26(4):938–948, 2014.

[173] Yuan Yao, Bin Xiao, Gaofei Wu, Xue Liu, Zhiwen Yu, Kailong Zhang, and Xingshe Zhou. Multi-channel based sybil attack detection in vehicular ad hoc networks using RSSI. *IEEE Transactions on Mobile Computing*, 18(2):362–375, 2018.

[174] Xia Feng, Chun-yan Li, De-xin Chen, and Jin Tang. A method for defensing against multi-source sybil attacks in VANET. *Peer-to-Peer Networking and Applications*, 10(2):305–314, 2017.

[175] Seung-Hyun Seo, Jongho Won, Salmin Sultana, and Elisa Bertino. Effective key management in dynamic wireless sensor networks. *IEEE Transactions on Information Forensics and Security*, 10(2):371–383, 2014.

[176] Tiziri Oulhaci, Mawloud Omar, Fatiha Harzine, and Ines Harfi. Secure and distributed certification system architecture for safety message authentication in VANET. *Telecommunication Systems*, 64(4):679–694, 2017.

[177] James T Curran and Ali Broumendan. On the use of low-cost IMUs for GNSS spoofing detection in vehicular applications. In *2017 International Technical Symposium on Navigation and Timing (ITSNT)*, pages 1–8. ENAC, 2017.

[178] Qian Wang, Zhaojun Lu, Mingze Gao, and Gang Qu. Edge computing based GPS spoofing detection methods. In *2018 IEEE 23rd International Conference on Digital Signal Processing (DSP)*, pages 1–5. IEEE, 2018.

[179] Ming-Chin Chuang and Jeng-Farn Lee. TEAM: Trust-extended authentication mechanism for vehicular ad hoc networks. *IEEE Systems Journal*, 8(3):749–758, 2014.

[180] Ahmed Arsalan and Rana Asif Rehman. Prevention of timing attack in software defined named data network with VANETs. In *2018 International Conference on Frontiers of Information Technology (FIT)*, pages 247–252. IEEE, 2018.

[181] Adrian Perrig, Ran Canetti, J Doug Tygar, and Dawn Song. The Tesla broadcast authentication protocol. *RSA CryptoBytes*, 5(2):2–13, 2002.

[182] Ahren Studer, Fan Bai, Bhargav Bellur, and Adrian Perrig. Flexible, extensible, and efficient VANET authentication. *Journal of Communications and Networks*, 11(6):574–588, 2009.

[183] Puguang Liu, Bo Liu, Yipin Sun, Baokang Zhao, and Ilsun You. Mitigating DoS attacks against pseudonymous authentication through puzzle-based co-authentication in 5G-VANET. *IEEE Access*, 6:20795–20806, 2018.

[184] Yingmo Jie, Mingchu Li, Cheng Guo, and Ling Chen. Dynamic defense strategy against DoS attacks over vehicular ad hoc networks based on port hopping. *IEEE Access*, 6:51374–51383, 2018.

[185] Ying Gao, Hongrui Wu, Binjie Song, Yaqia Jin, Xiongwen Luo, and Xing Zeng. A distributed network intrusion detection system for distributed denial of service attacks in vehicular ad hoc network. *IEEE Access*, 7:154560–154571, 2019.

[186] Adil Mudasir Malla and Ravi Kant Sahu. Security attacks with an effective solution for DoS attacks in VANET. *International Journal of Computer Applications*, 66(22), 2013.

[187] Mohammad Javad Faghihniya, Seyed Mojtaba Hosseini, and Maryam Tahmasebi. Security upgrade against RREQ flooding attack by using balance index on vehicular ad hoc network. *Wireless Networks*, 23(6):1863–1874, 2017.

[188] Seyed Mohammad Safi, Ali Movaghar, and Misagh Mohammadizadeh. A novel approach for avoiding wormhole attacks in VANET. In *2009 2nd International Workshop on Computer Science and Engineering*, volume 2, pages 160–165. IEEE, 2009.

[189] Shahjahan Ali, Parma Nand, and Shailesh Tiwari. Secure message broadcasting in VANET over wormhole attack by using cryptographic technique. In *2017 International Conference on Computing, Communication and Automation (ICCCA)*, pages 520–523. IEEE, 2017.

[190] Ameneh Daeinabi and Akbar Ghaffarpour Rahbar. Detection of malicious vehicles (DMV) through monitoring in vehicular ad-hoc networks. *Multimedia Tools & Applications*, 66(2):325–338, 2013.

[191] Raghad Baiad, Omar Alhussein, Hadi Otrok, and Sami Muhaidat. Novel cross layer detection schemes to detect blackhole attack against QoS-OLSR protocol in VANET. *Vehicular Communications*, 5:9–17, 2016.

[192] Zaid A Abdulkader, Azizol Abdullah, Mohd Taufik Abdullah, and Zuriati Ahmad Zukarnain. LI-AODV: Lifetime improving AODV routing for detecting and removing black-hole attack from VANET. *Journal of Theoretical & Applied Information Technology*, 95(1):1–15, 2017.

[193] Jin Cui, Lin Shen Liew, Giedre Sabaliauskaite, and Fengjun Zhou. A review on safety failures, security attacks, and available countermeasures for autonomous vehicles. *Ad Hoc Networks*, 90:101823–101836, 2019.

[194] Lynda Mokdad, Jalel Ben-Othman, and Anh Tuan Nguyen. DJAVAN: Detecting jamming attacks in vehicle ad hoc networks. *Performance Evaluation*, 87:47–59, 2015.

[195] Dimitrios Karagiannis and Antonios Argyriou. Jamming attack detection in a pair of rf communicating vehicles using unsupervised machine learning. *Vehicular Communications*, 13:56–63, 2018.

[196] Li He and Wen Tao Zhu. Mitigating DoS attacks against signature-based authentication in VANETs. In *2012 IEEE International Conference on Computer Science and Automation Engineering (CSAE)*, volume 3, pages 261–265. IEEE, 2012.

[197] Albert Wasef, Rongxing Lu, Xiaodong Lin, and Xuemin Shen. Complementing public key infrastructure to secure vehicular ad hoc networks. *IEEE Wireless Communications*, 17(5):22–28, 2010.

[198] Avleen Kaur Malhi and Shalini Batra. Genetic-based framework for prevention of masquerade and DDoS attacks in vehicular ad-hoc networks. *Security and Communication Networks*, 9(15):2612–2626, 2016.

[199] Wenjia Li and Houbing Song. ART: An attack-resistant trust management scheme for securing vehicular ad hoc networks. *IEEE Transactions on Intelligent Transportation Systems*, 17(4):960–969, 2016.

[200] Murtadha A Alazzawi, Hongwei Lu, Ali A Yassin, and Kai Chen. Efficient conditional anonymity with message integrity and authentication in a vehicular ad-hoc network. *IEEE Access*, 7:71424–71435, 2019.

[201] Nai-Wei Lo and Hsiao-Chien Tsai. Illusion attack on VANET applications—a message plausibility problem. In *2007 IEEE Globecom Workshops*, pages 1–8. IEEE, 2007.

[202] Jithin Zacharias and Sibylle Fröschle. Misbehavior detection system in VANETs using local traffic density. In *2018 IEEE Vehicular Networking Conference (VNC)*, pages 1–4. IEEE, 2018.

[203] Xiaoyan Zhu, Shunrong Jiang, Liangmin Wang, and Hui Li. Efficient privacy-preserving authentication for vehicular ad hoc networks. *IEEE Transactions on Vehicular Technology*, 63(2):907–919, 2013.

[204] Paolo Cencioni and Roberto Di Pietro. VIPER: A vehicle-to-infrastructure communication privacy enforcement protocol. In *2007 IEEE International Conference on Mobile Adhoc and Sensor Systems*, pages 1–6. IEEE, 2007.

[205] Canhuang Dai, Xingyu Xiao, Yuzhen Ding, Liang Xiao, Yuliang Tang, and Sheng Zhou. Learning based security for VANET with blockchain. In *2018 IEEE International Conference on Communication Systems (ICCS)*, pages 210–215. IEEE, 2018.

[206] Gilles Guette and Bertrand Ducourthial. On the sybil attack detection in VANET. In *2007 IEEE International Conference on Mobile Ad hoc and Sensor Systems*, pages 1–6. IEEE, 2007.

[207] Tassos Dimitriou, Ebrahim A Alrashed, Mehmet Hakan Karaata, and Ali Hamdan. Imposter detection for replication attacks in mobile sensor networks. *Computer Networks*, 108:210–222, 2016.

[208] Maria Azees, Pandi Vijayakumar, and Lazarus Jegatha Deboarh. EAAP: Efficient anonymous authentication with conditional privacy-preserving scheme for vehicular ad hoc networks. *IEEE Transactions on Intelligent Transportation Systems*, 18(9):2467–2476, 2017.

[209] Lina Bariah, Dina Shehada, Ehab Salahat, and Chan Yeob Yeun. Recent advances in VANET security: A survey. In *2015 IEEE 82nd vehicular technology conference (VTC2015-Fall)*, pages 1–7. IEEE, 2015.

[210] Timothy Trippel, Ofir Weisse, Wenyuan Xu, Peter Honeyman, and Kevin Fu. WALNUT: Waging doubt on the integrity of MEMS accelerometers with acoustic injection attacks. In *2017 IEEE European symposium on security and privacy (EuroS&P)*, pages 3–18. IEEE, 2017.

[211] Ajay Rawat, Santosh Sharma, and Rama Sushil. VANET: Security attacks and its possible solutions. *Journal of Information and Operations Management*, 3(1):301–304, 2012.

[212] Diego Kreutz, Fernando MV Ramos, Paulo Esteves Verissimo, Christian Esteve Rothenberg, Siamak Azodolmolky, and Steve Uhlig. Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, 103(1):14–76, 2014.

[213] Lixia Zhang, Alexander Afanasyev, Jeffrey Burke, Van Jacobson, KC Claffy, Patrick Crowley, Christos Papadopoulos, Lan Wang, and Beichuan Zhang. Named data networking. *ACM SIGCOMM Computer Communication Review*, 44(3):66–73, 2014.

[214] Sherali Zeadally, Ray Hunt, Yuh-Shyan Chen, Angela Irwin, and Aamir Hassan. Vehicular ad hoc networks (VANETs): Status, results, and challenges. *Telecommunication Systems*, 50(4):217–241, 2012.

[215] Henry CJ Lee and Vrizlynn L L Thing. Port hopping for resilient networks. In *2004 IEEE 60th Vehicular Technology Conference (VTC)*, volume 5, pages 3291–3295. IEEE, 2004.

[216] Kaishun Wang, Jun Guo, and Fenggao Li. Singular linear space and its applications. *Finite Fields and Their Applications*, 17(5):395–406, 2011.

[217] Irshad Ahmed Sumra, Halabi Bin Hasbullah, and Jamalul-lail Bin Ab-Manan. Attacks on security goals (confidentiality, integrity, availability) in VANET: A survey. In *Vehicular Ad-Hoc Networks for Smart Cities*, pages 51–61. Springer, 2015.

[218] Fatih Sakiz and Sevil Sen. A survey of attacks and detection mechanisms on intelligent transportation systems: VANETs and IoV. *Ad Hoc Networks*, 61:33–50, 2017.

[219] Mohamed Nidhal Mejri, Jalel Ben-Othman, and Mohamed Hamdi. Survey on VANET security challenges and possible cryptographic solutions. *Vehicular Communications*, 1(2):53–66, 2014.

[220] Vimal Bibhu, Kumar Roshan, Kumar Balwant Singh, and Dhirendra Kumar Singh. Performance analysis of black hole attack in VANET. *International Journal of Computer Network & Information Security*, 4(11):47–54, 2012.

[221] Dilendra Shukla, Akash Vaibhav, Sanjoy Das, and Prashant Johri. Security and attack analysis for vehicular ad hoc network—a survey. In *2016 International Conference on Computing, Communication and Automation (ICCCA)*, pages 625–630. IEEE, 2016.

[222] Rohini Rawat and Deepti Sharma. Impact of jamming attack in vehicular ad hoc networks. *International Journal of Advanced Research in Computer and Communication Engineering*, 4(4):457–461, 2015.

[223] Abderrahim Benslimane and Huong Nguyen-Minh. Jamming attack model and detection method for beacons under multichannel operation in vehicular networks. *IEEE Transactions on Vehicular Technology*, 66(7):6475–6488, 2016.

[224] Irshad Ahmed Sumra, Halabi Bin Hasbullah, Iftikhar Ahmad, and Daniyal M Alghazzawi. Classification of attacks in vehicular ad hoc network (VANET). *Information*, 16(5):2995–3004, 2013.

[225] José María De Fuentes, Ana Isabel González-Tablas, and Arturo Ribagorda. Overview of security issues in vehicular ad-hoc networks. In *Handbook of research on mobility and computing: Evolving technologies and ubiquitous impacts*, pages 894–911. IGI global, 2011.

[226] Ikram Ali, Alzubair Hassan, and Fagen Li. Authentication and privacy schemes for vehicular ad hoc networks (VANETs): A survey. *Vehicular Communications*, 16:45–61, 2019.

[227] Zhang Jianhong, Xu Min, and Liu Liying. On the security of a secure batch verification with group testing for VANET. *International Journal of Network Security*, 16(5):351–358, 2014.

[228] Debiao He, Sherali Zeadally, Baowen Xu, and Xinyi Huang. An efficient identity-based conditional privacy-preserving authentication scheme for vehicular ad hoc networks. *IEEE Transactions on Information Forensics and Security*, 10(12):2681–2691, 2015.

[229] Mohammed Saeed Al-Kahtani. Survey on security attacks in vehicular ad hoc networks (vanets). In *2012 6th International Conference on Signal Processing and Communication Systems*, pages 1–9. IEEE, 2012.

[230] Mushtak Y Gadkari and Nitin B Sambre. VANET: Routing protocols, security issues and simulation tools. *IOSR Journal of Computer Engineering*, 3(3):28–38, 2012.

[231] Muhammad Sameer Sheikh and Jun Liang. A comprehensive survey on VANET security services in traffic management system. *Wireless Communications and Mobile Computing*, 2019:1–23, 2019.

[232] Philippe Golle, Markus Jakobsson, Ari Juels, and Paul Syverson. Universal re-encryption for mixnets. In *2004 Cryptographers' Track at the RSA Conference*, pages 163–178. Springer, 2004.

[233] Bassem Mokhtar and Mohamed Azab. Survey on security issues in vehicular ad hoc networks. *Alexandria engineering journal*, 54(4):1115–1126, 2015.

[234] Network Simulator Version 2. http://www.isi.edu/nsnam/. [Online; accessed 05-July-2021].

[235] Network Simulator Version 3. https://www.nsnam.org. [Online; accessed 05-July-2021].

[236] OMNeT++. https://www.omnetpp.org. [Online; accessed 05-July-2021].

[237] Glomosim. https://networksimulationtools.com/glomosim/. [Online; accessed 05-July-2021].

[238] Eclipse Foundation. Simulation of Urban MObility (SUMO). https://www.eclipse.org/sumo/. [Online; accessed 05-July-2021].

[239] Jérôme Härri, Fethi Filali, Christian Bonnet, and Marco Fiore. VanetMobiSim: Generating realistic mobility patterns for VANETs. In *2006 3rd International Workshop on Vehicular Ad hoc Networks*, pages 96–97, 2006.

[240] Michal Piorkowski, Maxim Raya, A Lezama Lugo, Panagiotis Papadimitratos, Matthias Grossglauser, and J-P Hubaux. TraNS: Realistic joint traffic and network simulator for VANETs. *ACM SIGMOBILE Mobile Computing and Communications Review*, 12(1):31–33, 2008.

[241] MathWorks Inc. Matrix Laboratory (MATLAB). https://www.mathworks.com/products/matlab/. [Online; accessed 05-July-2021].

[242] Muhammad Baqer Mollah, Jun Zhao, Dusit Niyato, Yong Liang Guan, Chau Yuen, Sumei Sun, Kwok-Yan Lam, and Leong Hai Koh. Blockchain for the Internet of Vehicles towards intelligent

transportation systems: A survey. *IEEE Internet of Things Journal*, 8(6):4157–4185, 2021.

[243] Rajesh Gupta, Sudeep Tanwar, Neeraj Kumar, and Sudhanshu Tyagi. Blockchain-based security attack resilience schemes for autonomous vehicles in Industry 4.0: A systematic review. *Computers & Electrical Engineering*, 86:106717, 2020.

[244] Xifeng Wang, Changqiao Xu, Zan Zhou, Shujie Yang, and Limin Sun. A survey of blockchain-based cybersecurity for vehicular networks. In *2020 International Wireless Communications and Mobile Computing (IWCMC)*, pages 740–745. IEEE, 2020.

[245] Zhe Yang, Kan Yang, Lei Lei, Kan Zheng, and Victor C M Leung. Blockchain-based decentralized trust management in vehicular networks. *IEEE Internet of Things Journal*, 6(2):1495–1505, 2019.

[246] Chenyue Zhang, Wenjia Li, Yuansheng Luo, and Yupeng Hu. AIT: An AI-enabled trust management system for vehicular networks using blockchain technology. *IEEE Internet of Things Journal*, 8(5):3157–3169, 2021.

[247] Xiao Zheng, Mingchu Li, Yuanfang Chen, Jun Guo, Muhammad Alam, and Weitong Hu. Blockchain-based secure computation offloading in vehicular networks. *IEEE Transactions on Intelligent Transportation Systems*, 22(7):4073–4087, 2021.

[248] Ming Li, Jian Weng, Anjia Yang, Jia-Nan Liu, and Xiaodong Lin. Toward blockchain-based fair and anonymous Ad dissemination in vehicular networks. *IEEE Transactions on Vehicular Technology*, 68(11):11248–11259, 2019.

[249] Sowmya Kudva, Shahriar Badsha, Shamik Sengupta, Ibrahim Khalil, and Albert Zomaya. Towards secure and practical consensus for blockchain based VANET. *Information Sciences*, 545:170–187, 2021.

[250] Zhuo Ma, Junwei Zhang, Yongzhen Guo, Yang Liu, Ximeng Liu, and Wei He. An efficient decentralized key management mechanism for VANET with blockchain. *IEEE Transactions on Vehicular Technology*, 69(6):5836–5849, 2020.

[251] Yuling Chen, Xiaohan Hao, Wei Ren, and Yi Ren. Traceable and authenticated key negotiations via blockchain for vehicular communications. *Mobile Information Systems*, 2019:5627497, 2019.

[252] Kuljeet Kaur, Sahil Garg, Georges Kaddoum, François Gagnon, and Syed Hassan Ahmed. Blockchain-based lightweight authentication mechanism for vehicular fog infrastructure. In *2019 IEEE International Conference on Communications workshops (ICC workshops)*, pages 1–6. IEEE, 2019.

[253] California Department of Motor Vehicles. Autonomous vehicle disengagement reports. https://www.dmv.ca.gov/portal/vehicle-industry-services/autonomous-vehicles/disengagement-reports/, 2021. [Online; accessed 05-July-2021].

[254] Tesla Model S driver killed in Williston Florida. https://www.thecarcrashdetective.com/joshua-brown-tesla-model-s-driver-killed-williston-fl/, 2016. [Online; accessed 05-July-2021].

[255] Leonard Manson. Tesla autopilot makes Model 3 crash into overturned truck. https://www.somagnews.com/tesla-autopilot-makes-model-3-crash-overturned-truck/, June 2020. [Online; accessed 05-July-2021].

[256] Neville A Stanton, Paul M Salmon, Guy H Walker, and Maggie Stanton. Models and methods for collision analysis: A comparison study based on the Uber collision with a pedestrian. *Safety Science*, 120:117–128, 2019.

[257] National Transpotation Safety Board. Preliminary report highway: hwy18mh010. https://www.ntsb.gov/investigations/AccidentReports/Reports/HWY18MH010-prelim.pdf, 2018. [Online; accessed 05-July-2021].

[258] Andy Greenberg. Hackers remotely kill a jeep on the highway—with me in it. https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/, July 2015. [Online; accessed 05-July-2021].

[259] David Z Morris. Tesla-stealing hack is about much more than Tesla. https://fortune.com/2016/11/26/tesla-stealing-hack/, November 2016. [Online; accessed 05-July-2021].

[260] Denham Sadler. Cyber pro charged with GoGet hacking. https://ia.acs.org.au/article/2018/cyber-pro-charged-with-goget-hacking.html, February 2018. [Online; accessed 05-July-2021].

[261] Justine Sherry, Chang Lan, Raluca Ada Popa, and Sylvia Ratnasamy. Blindbox: Deep packet inspection over encrypted traffic. In *2015 ACM Conference on Special Interest Group on Data Communication (SIGCOMM)*, pages 213–226. ACM, 2015.

[262] Luca Deri and Francesco Fusco. Using deep packet inspection in cyber traffic analysis. In *2021 IEEE International Conference on Cyber Security and Resilience (CSR)*, pages 89–94. IEEE, 2021.

Cong Gao received the Ph.D. degree in computer architecture from Xidian University, Xi'an, China, in 2015. Currently, he is an Assistant Professor in the School of Computer Science and Technology at Xi'an University of Posts and Telecommunications, Xi'an, China. His current research interests include data sensing and fusion, autonomous driving, and network security.

Geng Wang received the B.S. degree in software engineering from Shanxi University, Taiyuan, China. He is currently pursuing the M.S. degree in software engineering at Xi'an University of Posts and Telecommunications. His current research interests include autonomous driving and anomaly detection of sensor data.

Weisong Shi received the Ph.D. in computer architecture from Chinese Academy of Sciences, Beijing, China, in 2000. He is a Fellow of IEEE, a Charles H. Gershenson Distinguished Faculty Fellow, and a Full Professor of Computer Science at the Wayne State University, USA. His current research interests include edge computing, computer systems for autonomous driving, mobile and connected health.

Zhongmin Wang received the Ph.D. degree in mechanical engineering and automation from the Beijing Institute of Technology, Beijing, China, in 2000. He is currently a Professor with the School of Computer Science and Technology, Xi'an University of Posts and Telecommunications. His current research interests include embedded intelligent perception, big data processing and application, and affective computing.

Yanping Chen received the Ph.D. degree in computer architecture from Xi'an Jiaotong University, Xi'an, China, in 2007. Currently, she is a Professor in the School of Computer Science and Technology at Xi'an University of Posts and Telecommunications, Xi'an, China. Her current research interests include service mining, service computing, and network management.