SAID: State-aware Defense Against Injection Attacks on In-vehicle Network

Lei Xue[1,2], Yangyang Liu[1], Tianqi Li[1], Kaifa Zhao[1],
Jianfeng Li[1], Le Yu[1], Xiapu Luo[1]*, Yajin Zhou[3], Guofei Gu[4]
[1]*The Hong Kong Polytechnic University*
[2]*The Hong Kong Polytechnic University Shenzhen Research Institute*
[3]*Zhejiang University* [4]*Texas A&M University*

Abstract

Modern vehicles are equipped with many ECUs (Electronic Control Unit) that are connected to the IVN (In-Vehicle Network) for controlling the vehicles. Meanwhile, various interfaces of vehicles, such as OBD-II port, T-Box, sensors, and telematics, implement the interaction between the IVN and external environment. Although rich value-added functionalities can be provided through these interfaces, such as diagnostics and OTA (Over The Air) updates, the adversary may also inject malicious data into IVN, thus causing severe safety issues. Even worse, existing defense approaches mainly focus on detecting the injection attacks launched from IVN, such as malicious/compromised ECUs, by analyzing CAN frames, and cannot defend against the higher layer MIAs (Message Injection Attacks) that can cause abnormal vehicle dynamics. In this paper, we propose a new state-aware abnormal message injection attack defense approach, named SAID. It detects the abnormal data to be injected into IVN by considering the data semantics and the vehicle dynamics and prevents the MIAs launched from devices connected to the vehicles, such as the compromised diagnostic tools and T-boxes. We develop a prototype of SAID for defending against MIAs and evaluate it using both real road data and simulation data. The experimental results show that SAID can defend against more than 99% of the network and service layer attack traffic and all state layer MIAs, effectively enforcing the safety of vehicles.

1 Introduction

Modern vehicles comprise many software-driven ECUs (Electronic Control Unit), sensors, and actuators [73], which implement the functionalities of the vehicle and are in charge of controlling the vehicle, such as ABS (Anti-lock Brake System), entertainment system, etc. They are connected to the IVN (In-Vehicle Network), such as CAN (Controller Area Network), LIN (Local Interconnect Network), or FlexRay [74]. Since CAN, the de facto standard, has been widely used in modern vehicles, we focus on CAN in this paper, but our methodology can be applied to other IVNs.

The attacks on the IVN or ECUs can seriously endanger driving safety. As more and more vehicles provide interfaces (e.g., OBD-II, T-Box, etc.) to communicate with the external entities (e.g., diagnostic devices, network, and sensors), an increasing number of adversaries exploit such new attack surfaces to attack the vehicles. For instance, it has been reported that attackers can remotely control a Jeep running on the highway at 70 miles per hour [14]. As another example, by exploiting the vulnerabilities in the aftermarket vehicle diagnostic tools, researchers demonstrated how to remotely control the vehicle's doors, windows, and mirrors [58]. Moreover, the Tesla Model S and X cars were hacked by security analysts, who remotely controlled the maneuvers [15]. All these attacks were launched by injecting specific data into the vehicle's IVN through its exposed interfaces.

To protect the IVN from MIAs (message injection attacks), existing countermeasures can be divided into two categories: *message authentication* and *intrusion detection* [43]. The former uses various cryptography mechanisms to exclude unauthorized commands [57, 69, 80, 89, 94, 97]. Unfortunately, it is not easy to deploy them in practice due to the limited computation resources of ECUs. Moreover, they cannot prevent the compromised ECUs from sending attack messages. The latter employs many features to capture abnormal attack patterns, such as the entropy of IVN [70], message intervals [64], message correlation/consistency [71], ECU fingerprints [29], communication characteristics [34], etc.

However, existing IDS (Intrusion Detection System) approaches have the following limitations and thus cannot effectively defend against MIAs. First, they focus on CAN frames at low IVN layer (e.g., bus-off attack [34]) and therefore miss the attack messages exploiting higher layer diagnostic services because such messages are usually transmitted by normal CAN frames. Second, they capture the abnormal data according to statistic characteristics without semantic and context information, and hence can be easily evaded. Third, since the existing solutions do not consider vehicle states, they cannot

*The corresponding author.

detect the attack messages injected during particular vehicle states, such as turning off turn signals while cornering.

In this paper, we propose a new approach, named SAID, to defend against MIAs. Following the defense in depth principle, SAID scrutinizes the incoming messages at three layers due to the multi-layer model adopted by IVN data transmission. SAID decides whether a well-formed incoming message is benign according to the safety consequence it causes in the current message context and vehicle states, and thus SAID takes into account both the message context and the vehicle states. More precisely, SAID first detects the abnormal CAN frames at the network layer. Then it assembles the passed CAN frames into diagnostic messages and identifies the abnormal ones at the service layer. After that, at the state layer, it determines whether the messages should be allowed according to their semantics and the vehicle states, which are estimated based on the vehicle dynamics models and specified rules. It is worth noting that, SAID is deployed between the IVN and the out-vehicle entities. Although it focuses on defending against the MIAs launched from out-vehicle entities, it can also detect the anomalies caused by in-vehicle ECUs/sensors or the unusual states under the control of drivers.

It is non-trivial to design SAID because of two major challenging issues. `C1`: Sophisticated attack messages may cause severe safety consequences in specific driving states and can easily evade the existing detection methods with the right formats. For example, a compromised OBD-II dongle can cause rollover or side slip through injecting acceleration messages continuously during cornering. Therefore, we need to estimate the dynamic vehicle states and further design proper policies to detect such injected well-formatted malicious messages, which requires the lightweight and timely driving state estimation approach. `C2`: Similar to the OSI (Open System Interconnection) model, the IVN also adopts a multiple-layer architecture, and thus the adversary can inject well prepared data at different layers for various attack purposes, such as injecting high priority CAN frames for DoS (Denial of Service) attack and diagnostic messages for fuzzy testing the security key. To effectively defend against the attacks launched at all these layers, we need to consider both the multiple-layer characteristic of IVN and the communication protocols used at different layers.

In this paper, we leverage various novel detection mechanisms and designs to address these challenges. Particularly, to approach `C1`, we propose a new lightweight approach to estimate the dynamic vehicle states timely using three vehicle dynamics models (§4) and further analyze the injected messages considering both the vehicle states and the message semantics. Meanwhile, we collect context information from both IVN and onboard sensors for state estimation. To address `C2`, we design specific abnormal message detection policies for each layer considering the communication characteristics (§5). Also, we take into account the dependencies between these layers and adopt a cross-layer architecture to manage and implement them.

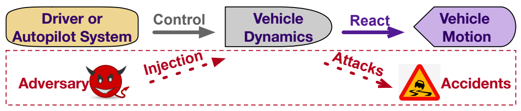


Figure 1: The control of vehicle dynamics.

Overall, we make the following major contributions.

- We propose a novel approach named SAID to defend against the MIAs from external entities. To the best of our knowledge, it is the *first* approach that considers both vehicle states and message semantics and conducts multi-layer screening to defend against sophisticated injection attacks.
- We develop a prototype of SAID and deploy it on real vehicles by solving a series of challenging issues, such as vehicle state estimation and cross-layer malicious message detection. Both code and datasets can be accessed at https://github.com/rewhy/said.
- We evaluate SAID using professional simulation software, real vehicles, robotic cars, and a well-designed testbed. The experimental results show SAID can effectively defend against the injection attacks to IVN. For example, SAID can prevent all state layer attacks and filter out more than 99% network and service layer attack traffic.

2 Background

This section briefly introduces the necessary background about vehicle dynamics, CAN, and diagnostic services.

Vehicle Dynamics: Vehicle Dynamics is defined as the application of the physical laws to a vehicle in motion [9]. As shown in Fig. 1, the vehicle dynamics models the reactions of the vehicle to the inputs from the drivers or the autopilot systems. That is, it studies how the motions of the vehicle are triggered by the actions of steering, accelerating, and braking [19]. Overall, the vehicle dynamics can be divided into lateral dynamics, longitudinal dynamics, and vertical dynamics, considering the abnormal dynamic vehicle states (i.e., accidents in Fig. 1) that can be prevented [99], such as side slip and rollover. In this paper, we aim to defend against MIAs by applying the vehicle dynamics models to checking whether the incoming messages will cause anomalies, and then the messages causing abnormal dynamic states are discarded.

Control Area Network (CAN): In a vehicle using CAN bus, many ECUs are connected to it and exchange messages for controlling the electrical modules of the vehicle, such as the engine control module (ECM) [48]. The communication follows the CAN protocol (ISO 11898 [40]), which specifies the first and the second layers of the OSI (Open System Interconnection) model. This is a frame-based protocol with a typical speed of 500 kbit/s [40]. The CAN frames are broadcast to all ECUs connecting to the CAN bus, and the ECUs just process the frames targeted to themselves and ignore others.

Diagnostic Services: The diagnostic services provide a set of functions, such as test, inspection, monitoring, diagnosis, or programming of the in-vehicle sub-systems. Besides the OBD (On-board diagnostic) services for the mandatory emissions-related diagnostics, there are two popular diagnostic services, namely UDS (Unified Diagnostic Services) and DoCAN (Diagnostic communication over CAN). Each diagnostic service involves at least three types of diagnostic messages, including requests, positive responses, and negative responses.

3 Overview

This section first presents our threat model and then gives an overview of our new defense approach against MIAs.

3.1 Threat Model

In this paper, we aim at defending against the attacks that inject abnormal data/messages into IVN from out-vehicle entities (e.g., telematics, OBD-II ports, T-Box, and sensors) [20, 23, 44, 58, 65, 96, 97, 102]. We focus on the MIA launched from out-vehicle entities with two goals.

Functional Attacks (MIA-❶): They aim to cause malfunctions on the IVN by injecting attack traffic, such as spoofing, DoS (Denial of Service), bus-off, and fuzzy attacks, which do not require the adversary has much knowledge about the IVN communication protocols [21, 26, 62, 75, 92, 100]. For instance, by exploiting the vulnerable error-handling scheme of CAN bus, the adversary can inject specific CAN frames to disconnect or shutdown the target ECU [28].

State Attacks (MIA-❷): They aim to cause abnormal vehicle states and motions, such as rollover, side slip, and turning without any turn signal. Such attacks always require the adversary has rich knowledge of the target vehicles because he/she needs to inject special messages into IVN to control the vehicle [14, 15, 58]. For example, by hacking the OBD-II port, an adversary can make a vehicle rollover by injecting well prepared messages into IVN to persistently accelerate while cornering [14, 15, 58]. Also, an adversary can let the vehicle enter abnormal states by spoofing the sensors (e.g., accelerator, GPS, and cameras) [22, 50, 84].

Assumptions: SAID is implemented on the specific hardware equipped with basic sensors (i.e., gyroscope and accelerometer). We assume both the software and hardware of SAID are reliable and the DBC file of the IVN is available, which provides a way to parse the IVN messages [101]. During dynamic state estimation, the vehicle is assumed to have planar motions, and the hardware of SAID locates at the center of gravity (*cg*) of the vehicle. The *cg* does not vary much with the changes of passengers in the vehicle because the weight of the passengers is much lighter than that of the vehicle. Since the vehicle parameters provided by the manufacturers contain the distances from the front wheel axle and rear wheel axle to the *cg* of the vehicle, denoted by l_f and l_r respectively, we can determine the *cg* of the vehicle.

Defense Scenario: SAID defends against the MIAs launched from out-vehicle entities. Although it does not prevent in-vehicle ECU/Sensor from transmitting data, it can detect the anomalies caused by them and also warn the driver. Consequently, if a suspicious/unusual state is controlled by the driver, SAID will warn the driver about it but not prevent it.

3.2 Defense Approach

To defend against the two types of MIAs, SAID is deployed between the IVN and the out-vehicle entities. SAID inspects all incoming data at three layers (i.e., network layer, service layer, and state layer) because IVN adopts a multi-layer model with diverse protocols at each layer.

At *network layer* and *service layer*, SAID defends against MIA-❶ through inspecting the incoming CAN frames and diagnostic messages, respectively. More precisely, SAID scrutinizes the incoming data, including CAN frames and diagnostic messages, according to the protocol specifications at network layer and service layer, respectively. Meanwhile, SAID also detects functional attacks according to the specified detection rules for these two layers.

At *state layer*, SAID defends against MIA-❷ by estimating the vehicle states and parsing the semantics of the incoming messages. Particularly, for the injected well-formatted messages, SAID inspects whether they will trigger abnormal vehicle states, including dangerous vehicle motions (e.g., rollover and slide slip) and abnormal driving behaviors (i.e., opening doors on the highway and turning off signals during cornering). We estimate the vehicle states leveraging the vehicle dynamics models and further defend against MIAs using both rule-based and dynamics-based defense policies.

It is worth noting that vehicle manufacturers usually endeavor to enhance the vehicle's dynamic safety. In the trend of 'control-by-wire' and autonomous driving, many prior works demonstrated attacks that can take control of the vehicle. Therefore, it is also important to protect the vehicle's dynamic states from the cyber-security aspect, and furthermore the dynamic safety is also the last line of defense for cyber-security. Our dynamics-based method focuses on the detection of attack messages that are injected to trigger dangerous dynamic states, such as the vehicle state of 'loss of control'. Hence, we estimate the dynamic states of the vehicle and further utilize representative indices of the dynamic states to indicate the dynamic risks.

L1. Network Layer: In IVN (i.e., CAN), all data is transmitted in CAN frames at network layer and thus SAID detects abnormal CAN frames at this layer according to the policies derived from the protocol specifications, research papers, and technical reports. More precisely, SAID first checks whether the incoming CAN frames have legitimate formats and are transmitted following the protocol specifications [1]. For the legitimate frames, SAID further inspects whether they are injected for specific network attacks, such as DoS attacks and fuzzy attacks, according to their transmission pattern [26, 52, 61, 70, 85]. If the CAN frames pass all network layer detection rules, SAID assembles them into diagnostic messages following the single-frame or multiple-frame transmission schemes and then delivers them to the service layer for further inspection.

L2. Service Layer: The service layer involves various diagnostic messages to transmit requests and receive responses. We prepare defense policies according to the protocol specifications (i.e., ISO 15031-5 [7], ISO 14229-1/2/3 [4–6], and ISO 15765-3/4 [2, 3]), and then apply them to detect attack messages violating the prescribed formats and transmission protocols (§5.2). The messages passing the service layer policies are further inspected at the state layer.

L3. State Layer: At state layer, SAID inspects the incoming messages considering both their semantics and the vehicle states. If the incoming message will let the vehicle enter an abnormal state, SAID will refuse it and raise a warning. The states estimated by SAID include both the vehicle motions (e.g., roll, steering, and accelerating/braking) and the status of the vehicle components (e.g., windows, signals, and doors).

During vehicle state estimation, we continuously retrieve the context information from onboard sensors and IVN via the diagnostic services, and leverage them to identify the current vehicle motions following the vehicle dynamics models, including roll dynamics, steering dynamics, and accelerating/braking dynamics. During defense, we parse the semantics of the incoming message and then determine whether it is allowed to enter into IVN based on its semantics and the vehicle states. Particularly, if a message will let the vehicle enter dangerous motions or violate any predefined driving policies, SAID discards it and raises a warning. For instance, if a message will increase steering angle when the vehicle is already in a dangerous roll state, it will be dropped, and meanwhile, the driver will be informed of the details with a warning message. If the messages are gradually injected to cause anomalies not exceeding thresholds, the dynamics-based defense does not refuse them because no dynamics-based policy is violated, but they could be detected by other defense policies (e.g., network/service layer defense).

We design the vehicle state estimation algorithms based on vehicle dynamics models (§4) and construct the state layer defense policies by surveying the economic driving suggestions [11–13], safe driving requirements [10, 16, 17], as well as the auto insurance pricing policies [55, 59, 83] (§5.3).

Synergies: The state-aware detection provides state information for rule-based (i.e., network and service layer) defense, which makes decisions considering the vehicle states. Meanwhile, the rule-based defense module extracts message semantics for state-aware detection, thereby improving its effectiveness. Also, SAID collects onboard context information (accelerations and angular velocities) and in-vehicle context information (speed, steering angles, etc.) for state-aware detection, which uses the context information to estimate the vehicle dynamic states and further leverage such state information to decide whether the messages that have passed the network defense policies can be sent to IVN.

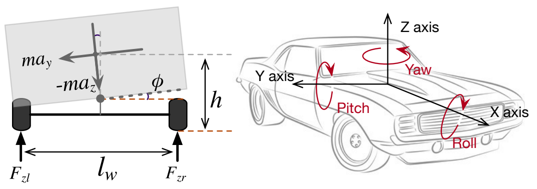


Figure 2: Vehicle rollover dynamics and axis system.

4 Vehicle Dynamics Model

We focus on the vehicle dynamics involving roll, steering, and accelerating/braking, which closely depends on the control inputs related to throttle, steering wheel angle, and brake, to detect the relevant attack messages. This section presents three vehicle dynamics models, including roll dynamics (`RD`), steering/yaw dynamics (`SD`), and accelerating/braking dynamics (`BD`), as well as the criteria for anomaly detection.

4.1 Roll Dynamics (RD)

The term *rollover index* is used to indicate the real-time likelihood of the vehicle to rollover in vehicle roll dynamics. To defend against the MIAs that cause rollovers, we calculate the *rollover index* using the real-time difference in vertical tire loads between left and right sides of the vehicle [81], which is also known as vertical load transfer ratio (LTR). The right subfigure of Fig. 2 shows the vehicle coordinate, where we use a_x, a_y, and a_z to represent the accelerations on the x-axis, y-axis, and z-axis directions. The left subfigure depicts the schematic of a vehicle with a sprung mass that undergoes roll motion, where h, ℓ_w, and ϕ denote the height from ground to the roll center of the vehicle, the track, and the roll angle, respectively. Eq. (1) is used to calculate the LTR based on the difference between the vertical tire forces F_{zr} and F_{zl}, which are caused by the roll motion.

$$LTR = \frac{F_{zl} - F_{zr}}{F_{zl} + F_{zr}} \tag{1}$$

As shown in Fig. 2, the vertical forces on the left tires (i.e., F_{zl}) increase while the vertical forces on the right tires (i.e., F_{zr}) decrease when the vehicle has a positive roll angle (i.e., $\phi > 0$), and hence right tires will lift off the ground (i.e., $F_{zl} = 0$) when ϕ is large. Likewise, if ϕ has a significant negative value, the left tires will lift off the ground (i.e., $F_{zr} = 0$). Consequently, the tire lift-off happens when $F_{zl} = 0$ or $F_{zr} = 0$, corresponding to $LTR = 1$ or $LTR = -1$, respectively, and therefore LTR is in range $(-1, 1)$ to prevent rollover.

The calculation of LTR is not straightforward because we cannot obtain F_{zl} and F_{zr} directly. Since the left and right tires are two support points, the vehicle lateral dynamics can be characterized by Eq. 2 and 3 approximately according to torque equilibrium. By combining Eq. 1, 2, and 3, LTR can be calculated through Eq. 4.

$$F_{zr} \cdot \ell_w + ma_y \cdot h \cdot \cos\phi + ma_z \cdot \frac{\ell_w}{2} \cdot \cos\phi = 0 \quad (2)$$

$$F_{zl} \cdot \ell_w - ma_y \cdot h \cdot \cos\phi + ma_z \cdot \frac{\ell_w}{2} \cdot \cos\phi = 0 \quad (3)$$

$$LTR = \frac{F_{zl} - F_{zr}}{F_{zl} + F_{zr}} = \frac{2a_y h}{a_z \ell_w} \quad (4)$$

Criterion: For vehicle roll stability and early warning, the LTR should be kept in a special range. That is, a_y and a_z should meet the requirement represented by Eq. 5, where LTR_{ro} is the threshold for safe roll motions, and its default value is 0.6 according to [56]. Thus, when Eq. 5 is not met, the vehicle has a high risk of rollover. Moreover, the risk increases with the increments of vehicle speed v_a and steering angle δ because a_y depends on them. When $|LTR|$ is 1, the tires of one side will lift off the ground.

$$\left|\frac{a_y}{a_z}\right| < \frac{\ell_w}{2h} \cdot LTR_{ro} \quad (5)$$

4.2 Steering (Yaw) Dynamics (SD)

An adversary can cause steering dynamics, such as severe under/oversteer, during cornering by injecting specific messages [8]. Particularly, if the wheels of the vehicle begin to side slip (i.e., loss grip) when cornering, the vehicle is in understeer or oversteer. In practice, the low under/oversteer is common and not noticed by the drivers, and the severe under/oversteer can let feel "loss of vehicle control". Also, when severe under/oversteer occurs, the actual tire track will be obviously different from the driver's intended track, and such a state is difficult to correct. Raimondo et al. [82] analyzed 1,674 accidents in 5 European countries from EACS (European Accident Causation Survey) and showed that many accidents causation was identified as "loss of vehicle control". In particular, the causation of around 50% of injury accidents occurring in cornering and around 18% of injury accidents occurring in straight lines is because the driver loses control of his/her vehicle, either laterally (i.e., slide slip) or longitudinally (e.g., wheels are locked during braking).

To defend against such attacks, we leverage steering dynamics to identify the incoming messages that can cause oversteer or understeer, which is depicted by the right part of Fig. 3. Specifically, understeer indicates that the required grip exceeds the traction of the front tires and the vehicle head slides wide across the ground; Oversteer means that the required grip exceeds the traction of the rear tires, and the vehicle tail slides wide.

$$\delta_i = \arctan\left(\frac{L}{r - \frac{\ell_w}{2}}\right) \quad (6)$$

$$\delta_o = \arctan\left(\frac{L}{r + \frac{\ell_w}{2}}\right) \quad (7)$$

The vehicles use the Ackermann steering geometry, and the front tires are the steering wheels [53]. As shown in the left part of Fig. 3, during cornering, the front inner tire steers for a

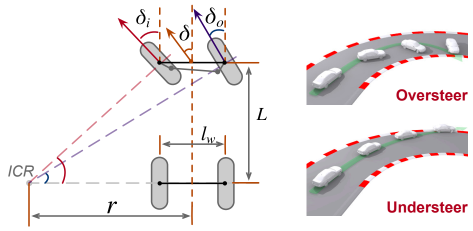


Figure 3: Ackermann steering geometry and under/oversteer.

bigger angle than the front outer tire in order to let the vehicle rotate around the middle point between the rear tires, and there exists a common point, called the instantaneous center of rotation (`ICR`), lying in the intersection of all tire axes. In Fig. 3, we use δ_i, δ_o, and δ to represent the steering angles of the front inner tire, the front outer tire, and the vehicle heading direction, respectively. L and r are the wheelbase and the turning radius of vehicle, respectively, and ℓ_w is the track.

During cornering, the steering angles can be determined by Eq. 6 and 7. Since L and ℓ_w are small compared to r, these two equations can be approximated by Eq. 8 and 9, and then the neutral steering angle of the vehicle $\delta_{neutral}$ can be obtained by Eq. 10. Note that the neutral steering angle $\delta_{neutral}$ is the required steering angle to turn around the `ICR` if there is no side tire slip [49]. Understeer occurs if $|\delta| < |\delta_{neutral}|$, where δ is the actual vehicle steering angle and obtained through dividing the steering wheel angle by steering wheel ratio δ_w/λ. Inversely, oversteer occurs if $|\delta| > |\delta_{neutral}|$.

$$\delta_i \approx \frac{L}{r - \frac{\ell_w}{2}} \quad (8)$$

$$\delta_o \approx \frac{L}{r + \frac{\ell_w}{2}} \quad (9)$$

$$\delta_{neutral} \approx \frac{\delta_i + \delta_o}{2} \cong \frac{L}{r} \quad (10)$$

If the cornering speed is constant, the centripetal acceleration a_y of the vehicle is calculated by Eq. 11, and we can further get Eq. 12 by substituting Eq. 10 and 11.

$$a_y = \frac{{V_x}^2}{r} \quad (11)$$

$$\delta_{neutral} \approx \frac{L \cdot a_y}{{V_x}^2} \quad (12)$$

Criterion: It is worth noting that small understeer or oversteer is common. To prevent abnormal steering motions, we regard the cases when δ is larger than $1.2 \cdot |\delta_{neutral}|$ and smaller than $0.8 \cdot |\delta_{neutral}|$ as severe oversteer and severe understeer, respectively. When severe oversteer occurs, the driver needs to reduce the throttle or brake as well as the steering wheel angle, but the driver should not suddenly change the throttle, brake, or steering wheel angle when the understeer occurs [38]. Therefore, the increases of throttle, brake, and steering wheel angle are abnormal in the oversteer state, and any quick change of them is abnormal in the understeer state.

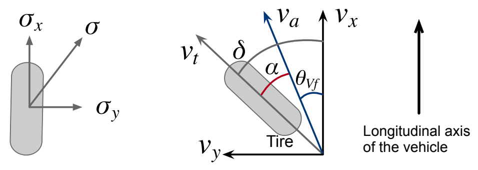


Figure 4: Vehicle tire slip geometry.

4.3 Accelerating/braking Dynamics (BD)

Tire slip occurs when accelerating and braking, and it represents the difference between the actual speed at the axle of the vehicle and the rotational speed of the tire. The total slip S consists of longitudinal slip and lateral slip [49], denoted by S_x and S_y, respectively. The total slip ratio σ is the vector sum of longitudinal slip ratio σ_x and lateral slip ratio σ_y, as shown in Eq. 13.

$$\sigma = \sqrt{\sigma_x^2 + \sigma_y^2} \tag{13}$$

σ_x denotes the ratio of the difference between the forward velocity of the vehicle V_x and the tire revolution. It is computed by Eq 14, where ω_w is the angular velocity of wheel and r_{eff} is the effective radius of the corresponding free-rolling tire.

$$\sigma_x = \begin{cases} \dfrac{r_{eff}\omega_w - V_x}{V_x} & accelerating \\ \dfrac{r_{eff}\omega_w - V_x}{r_{eff}\omega_w} & breaking \end{cases} \tag{14}$$

The lateral slip ratio σ_y is defined by Eq. 15, where α is the slip angle of the tire as shown in Fig. 4. For front tires, α_f is equal to the steering angle δ minus the front tire velocity angle θ_{Vf} (i.e., Eq. 16). For the rear tires, α_r is equal to the negative of θ_{Vf} (i.e., Eq. 17).

$$\sigma_y = \frac{V_x}{r_{eff}\omega_w} \tan\alpha \tag{15}$$

$$\alpha_f = \delta - \theta_{Vf} \tag{16}$$

$$\alpha_r = -\theta_{Vf} \tag{17}$$

θ_{Vf} equals the arctangent of the sum of lateral vehicle speed V_y and the lateral velocity because the yaw motion of the vehicle $\ell_f\dot{\psi}$ is divided by the longitudinal vehicle speed V_x. $\dot{\psi}$ is the rate of change of orientation of the vehicle, and it approximately equals the z-axis angular velocity (i.e., $\dot{\psi} \approx \omega_z$). ℓ_f is the distances of front wheel axle to the cg. Consequently, θ_{Vf} can be approximated using small angle approximations as following

$$\theta_{Vf} = \frac{V_y + \ell_f\dot{\psi}}{V_x} \tag{18}$$

Since the change of V_x during accelerating/braking approximates to the integral of longitudinal acceleration over time, we can obtain V_x using Eq. 19-20, where P is the frequency of reading data from onboard sensors, $\hat{V}_x$ and $\hat{V}_y$ are the x-axis speed and y-axis at the beginning of accelerating or braking.

$$\hat{V}_x + \int_0^t a_x \cdot dt = \hat{V}_x + \sum a_x \cdot \frac{1}{P} \tag{19}$$

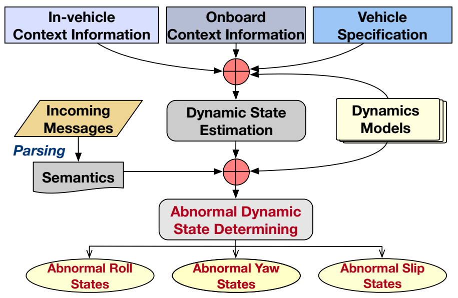


Figure 5: The vehicle dynamics state estimation approach.

$$\hat{V}_y + \int_0^t a_y \cdot dt = \hat{V}_y + \sum a_y \cdot \frac{1}{P} \tag{20}$$

Consequently, we obtain σ through Eq. 14-18 and the data from IVN, onboard sensors, and the vehicle specifications.
***Criterion**:* The tire force is proportional to the slip ratio when the slip ratio is small and inverse proportional to the slip ratio when the slip ratio increases [81]. Literature suggests the maximum force of a tire happens when the σ is between 0.1 to 0.2 [81, 91], and our simulations also show 0.2 gives the best performance. Therefore, we set the threshold of the tire slip ratio to 0.2, and any incoming message causing σ to exceed it will be rejected as an anomaly. Note that if σ is 1, it means the tire is completely held still and locked, in other words, the tire loses traction.

4.4 Dynamic State Estimation

Both in-vehicle and onboard context information is required to estimate the dynamic vehicle states.
***In-vehicle Information Collection**:* We collect the in-vehicle information using the diagnostic services. Specifically, we construct and send diagnostic requests to the IVN and parse the replied diagnostic responses based on the DBC file, which describes how to decode the fields in diagnostic messages.
***Onboard Information Collection**:* Although, for some vehicles, the gyroscope and accelerator information can be retrieved from the IVN, SAID reads such information from the onboard sensors because of two major reasons. First, the vehicle sensors can be compromised by the adversary potentially. Second, if using in-vehicle data, an active non-default (i.e., extended Diagnostic Session [5]) diagnostic session needs to be maintained for retrieving the gyroscope and accelerator information all the time, which can impact the normal diagnostic sessions between the in-vehicle ECUs and the OBD dongles. By contrast, the onboard sensors provide gyroscope and accelerator information directly and timely. Moreover, such onboard sensor data can be used to verify the data retrieved from the IVN for anomaly detection. For instance, if malfunctions occur in the IVN sensors (e.g., lateral and vertical acceleration from onboard sensors), we can identify such errors based on the onboard sensor data.
***Data Preprocessing**:* As onboard sensors may be affected by some environmental factors, e.g., placement of sensors and vibration, we preprocess the sensor data through the following

Table 1: The summary of the context data and vehicle parameters that are used in this paper.

Data source	Symbol	Semantics
In-vehicle	V_x/V_y	the x/y-axis velocities of the vehicle (m/s)
	a'_x, a'_y, a'_z	the x/y/z-axis vehicle accelerations (m/s^2)*
	$\omega'_x, \omega'_y, \omega'_z$	the x/y/z-axis angular velocities (rad/s)*
	δ_w	the steering wheel (degree)
	ω_{rpm}	the engine speed (RPM)
	u	the gear position of the vehicle engine
Onboard	a_x, a_y, a_z	the x/y/z-axis vehicle accelerations (m/s^2)
	$\omega_x, \omega_y, \omega_z$	the x/y/z-axis angular velocities (rad/s)
Vehicle Specification	ℓ_f, ℓ_r	the distances of front/real wheel axle to the cg
	L, ℓ_w	the wheelbase and wheel track of vehicle (m)
	m	the weight of vehicle (kg)
	λ	the steering wheel ratio

* Not all vehicles provide these in-vehicle sensor data.

steps to make SAID resilient to such environmental factors.

S-❶ *Coordinate Calibration:* We transform coordinate calibration into a linear optimization problem and derive the optimal rotation matrix to align the coordinate of the onboard sensors with vehicle coordinate. We build the accurate coordinate calibration using the algorithm described in [86].

S-❷ *Data Denoising:* We employ Kalman Filter [25, 95] to filter out the noise of the sensor readings, which are incurred by device vibration and sensor sensitivity.

S-❸ *Bias Elimination:* We centralize the onboard sensor readings by subtracting their average value when the vehicle stops. In this way, we eliminate the deviations.

We do not use the state estimation based on computer vision because they require powerful computation resources and their accuracy depends on visibility [27]. For example, they may be affected by environment, such as weather and light. Since the GNSS (Global Navigation Satellite System) information is neither always available nor timely [68], especially when the vehicle runs in the urban area, we read the speed information from IVN instead of the onboard sensors. In addition, the onboard sensor data can also help to verify the corresponding data retrieved from the IVN.

Vehicle Motion Estimation: As shown in Fig. 5, we obtain the required information from three sources, including in-vehicle network, onboard sensors, and specifications, which are summarized in Table 1. Furthermore, we apply the dynamics models `RD` (§4.1), `SD` (§4.2), and `BD` (§4.3) to estimate the dynamic vehicle states, including roll states, yaw (steering) states, and tire slip states. Afterward, SAID checks whether the incoming message will cause abnormal dynamic states based on the estimated dynamic states, and the estimation details are presented in §5.3.

5 Injection Attack Defense

Injection attacks against the vehicle can be launched at all layers. For example, the adversary can inject high-priority CAN frames to launch DoS attacks at network layer or diagnostic requests with various parameters to conduct the fuzzy attack at the service layer [45, 54]. Therefore, to prevent injection attacks, SAID analyzes the incoming traffic at multiple layers, including network layer, service layer, and state layer.

5.1 Network Layer Defense

For many vehicles, such as Ford Escape [64] and Jeep Cherokee [66], the CAN frames can be directly injected into IVN through OBD-II port [45, 54], and thus the adversary can launch injection attacks on the IVN by sending CAN frames directly. To defend against the network layer injection attacks, SAID detects the injected abnormal CAN frames according to two major types of policies, which are related to the formats of CAN frames and the patterns of attack traffic, respectively. In particular, at this layer, we build 88 defense policies.

5.2 Service Layer Defense

At the service layer, we focus on detecting the diagnostic messages that violate the specifications of the diagnostic services, which specify both the diagnostic messages and the diagnostic sessions. Currently, SAID supports three types of diagnostic services (i.e., OBD, UDS, and KWP2000).

The diagnostic session is the basis of the communication between OBD-II port and in-vehicle ECUs, and the target ECU is in a specific diagnostic status during diagnosing. Given a diagnostic message, SAID first checks its format according to the defense policies derived from the specifications [2, 4, 5, 7]. If the diagnostic message's format contains errors, it is abnormal and dropped by SAID. This layer contains 504 default defense policies and also supports customized settings.

5.3 State Layer Defense

Since MIAs can be launched to cause abnormal driving states, SAID needs to determine the purposes (i.e., functionalities) of the diagnostic requests to prevent them from causing abnormal driving states. Currently, SAID prevents abnormal vehicle motions and states by leveraging the vehicle dynamics models (i.e., dynamics-based defense) and the rules extracted from safe/eco driving requirements and auto insurance policies (i.e., rule-based defense).

Dynamics-based Defense: Based on the dynamics models presented in §4, we propose 18 defense policies to prevent the MIAs that aim at causing abnormal vehicle motions. For instance, with policy `D18`, SAID drops the message that increases the steering wheel angle when the vehicle is in a risky tire slip state that is determined using the `BD` model.

Rule-based Defense: We summarize 37 abnormal states according to the safe driving rules [10, 16–18], the economic driving suggestions [11–13], and the auto insurance pricing policies [47, 55, 59, 83]. For instance, if an incoming message will turn on the turn signal when the vehicle runs straight, it is abnormal and dropped. We design 337 defense policies to defend against the injection attacks that potentially cause abnormal driving states.

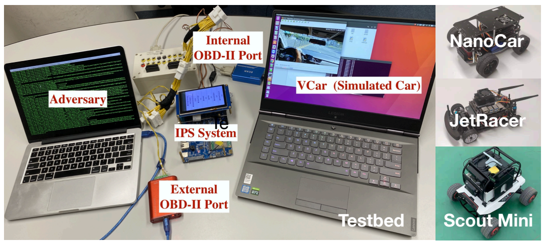


Figure 6: The topology of the testbed and robotic cars.

6 Evaluation

We implement a prototype of SAID on an iMx.6 board extended with sensors and two OBD-II ports, running an Android operating system, and the backend services run on the server, through which the users can update the defense rules. For the evaluation, we also implement a customized OBD-II dongle to collect CAN messages from real vehicles and inject CAN messages. As shown in Fig. 6, besides simulating various driving scenarios, we design experiments on the real vehicles, robotic cars, and testbed to evaluate SAID.

Simulation: To evaluate our dynamics-based state estimation and attack defense approaches, we use the professional vehicle simulator CarSim [63] to simulate various abnormal vehicle motions, including rollover, under/oversteer, and accelerating/braking, which are caused by injecting the messages related to throttle, steering wheel, and brake pedal, due to the ethics and safety considerations. We also evaluate the impacts of the model configurations (i.e., thresholds) on the performance of anomaly detection. More defense evaluations are presented in Appendix-A.1.

Robotic Cars: We also use three robotic cars (i.e., NanoCar, JetRacer Pro, and Scout Mini) to evaluate the performance of dynamics-based defense. It is worth noting that the scaled-down robotic cars have various differences from the real passenger vehicles, for example, these robotic cars have no brake, as well as smaller sizes and motion ranges. Consequently, they are more prone to environmental disturbance than a real passenger car. During the evaluation, we add extra loads to the robotic cars to make them more like real cars.

Real Vehicles: We conduct evaluations on seven real vehicles in total. First, we collect over 600 km real road IVN and diagnostic traffic from two real cars, a Skoda Octavia and a Volkswagen Lavida, driven by three drivers with video recording. The real-vehicle traffic is replayed on the testbed for evaluations in §6.2.2 and §6.3. We also evaluate SAID by launching functional service layer attacks on five additional cars, including Audi A4, BMW 523LI, Toyota Corolla HV, Nissan Teana, and Volkswagen Passat, and more details are presented in §6.2.2.

Testbed: We design and implement a vehicle *testbed*, named `VCar`, for evaluating SAID. It can replay the real-vehicle traffic and meanwhile play the recorded video in real-time. The topology of our testbed is shown in Fig. 6, where the vehicle is emulated by `VCar` and the PoC (proof-of-concept) attacks are launched by another laptop (i.e., Adversary). In other words, we mimic the adversary to launch attacks against the vehicle by injecting special data.

We evaluate SAID by answering four research questions.

- **RQ1:** Can SAID correctly estimate vehicle states before anomalies occur based on the vehicle dynamics models?
- **RQ2:** Can SAID effectively detect and prevent the state attacks (MIA-❶) by dropping attack messages?
- **RQ3:** Can SAID effectively detect and prevent the functional attacks (MIA-❷) by filtering out the attack data?
- **RQ4:** How is the additional data transmission latency introduced by SAID?

6.1 Abnormal Dynamic State Detection

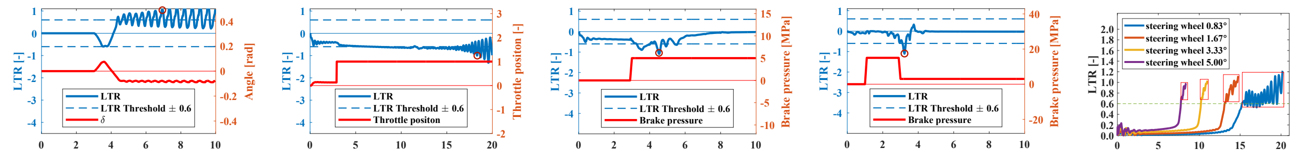
We evaluate whether SAID can correctly estimate the dynamic vehicle states before the anomalies (i.e., rollover and severe side slip) occur by simulating three major types of vehicle motions, including cornering, accelerating, and braking, using CarSim.

6.1.1 Roll-based Anomaly Detection: To evaluate whether SAID can correctly estimate the abnormal roll states, as shown in Fig. 7, we simulate various rollover motions, which are caused by different reasons, including steering angle, throttle inputs (acceleration), and brake pedal inputs (deceleration).

In Fig. 7(a), the rollover accident is due to the abnormal steering angle inputs. At the beginning, the vehicle is running at a stable speed of 80 km/h. From 3s, the diagnostic messages are injected to increase the steering wheel angle (left rotation is positive). Then the diagnostic messages are injected to decrease the steering wheel angle suddenly. As a result, the actual LTR exceeds LTR_{ro} from around 4.5s and further increases to around 1 at 7s. Note that $LTR = 1$ means the weight of the vehicle is fully transferred to the tires of one side and rollover may happen. In Fig. 7(b), from 3 s, the specific diagnostic messages (i.e., attack messages) are injected to increase the throttle from 0 to 1 and so that the vehicle speed is accelerated continuously, and then LTR exceeds 1 at around 18s. In Fig. 7(c) and Fig. 7(d), the specific diagnostic messages (i.e., attack messages) are injected to increase and decrease the brake pressure (pedal position) from 3s, and thus the vehicle speed increases and decreases correspondingly. Then LTR exceeds 1 at 4.5s in Fig. 7(c) and 3.5s in Fig. 7(d) due to the injected brake pressure in/decrease messages.

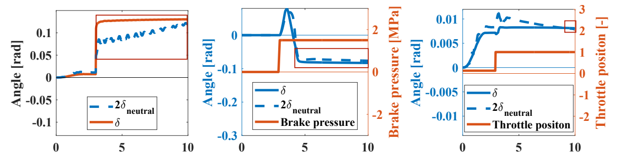
In addition, from Fig. 7, we can see that all abnormal vehicle states are correctly and timely detected by SAID before the rollovers occur using the `RD` model in §4.1.

We further evaluate the `RD` model with various threshold settings. As shown in Fig. 7(e), four rollover scenarios are simulated by continually accelerating the vehicles at full throttle under the constant steering angles of 0.83°, 1.67°, 3.33° and 5°, respectively. Meanwhile, the LTR increases as the vehicle speed grows until rollover occurs. The experimental results show that LTR rises rapidly after 0.2 and starts oscillating



(a) Steering angle (b) Throttle (c) Brake (Increase) (d) Brake (Decrease) (e) Threshold Sensitivity

Figure 7: The perception of rollover due to the injected messages of changing steering angle, throttle, and brake, as well as the simulated four rollover scenarios. The red circles (i.e., $LTR = 1$) in the left four figures indicate the tires of one slide are off the ground. In initial state: (a) $\delta_w = 0°$, in (b-d) $\delta_w = 55°$. The black boxes (right figure) indicate the periods of unstable vehicle states.



(a) Steering angle (b) Brake (c) Throttle

Figure 8: The perception of under/oversteer due to the injected messages that change steering angle, throttle, and brake. In the initial state, the δ_w are (a) (b) and (c) are 8°, 95° and 20°, the vehicle speed of (a) (b) and (c) are 60, 80 and 80 km/h, and road friction coefficient is 0.5. The red boxes indicate severe abnormal under/oversteer.

after 0.6, indicating that the vehicle is in an unstable state (annotated by the black boxes), and then rollover occurs after 5.0s, 1.7s, 0.8s, and 0.7s in these four scenarios, respectively. Hence, we set the default threshold of LTR to 0.6 considering both the unnecessary warnings and reaction time. SAID also allows users to set customized thresholds for their own vehicles. This default threshold (i.e., 0.6) is also used by the existing vehicle safety studies [36, 56].

6.1.2 Steering-based Anomaly Detection: In this experiment, we evaluate if SAID can detect abnormal steering motions before severe anomalies occur (e.g., under/oversteer). As shown in Fig. 8(a), there are diagnostic messages injected to change δ to 0.13 rad ($\delta_w = 150°$) from 3s. Thus, it will let the vehicle enter a severe under/oversteer state, if we consider the states that the ratio of the actual steering angle δ and the neutral steering angel $\delta_{neutral}$ is greater than a threshold (i.e., $\delta/\delta_{neutral} > 1.2$) as anomalies. Similarly, if we inject messages to change the brake pressure and throttle position, the vehicle also enters severe abnormal steering states, which are shown in Fig. 8(b) and Fig. 8(c), respectively. From state estimation results shown in Fig. 8, we can see that SAID can detect the abnormal under/oversteer states before severe anomalies occur by leveraging the SD model in §4.2.

To evaluate the impacts of various thresholds on the detection of abnormal steering states, we simulate the over/understeering motions under different settings (i.e., steering angle and speed). More precisely, in this test, we evaluate the actual values of the turn radius (r) and $\delta/\delta_{neutral}$ when the vehicle moves under constant steering angles and speeds. Note that the theoretical turn radius for setting steering angles, $\delta = 1.67°$, 5°, and 7°, are 100 m, 33.3 m, and 23.8 m, respectively, and the actual results are shown in Table 2. Note that,

Table 2: The actual turn radius of a vehicle when it turns under various constant steering angle r and speed settings. Turning radius r of various threshold setting for the yaw model, $\delta = 1.67°$, 5°, and 7°, the theoretical turn radius are 100.0 m, 33.3 m, and 23.8 m, respectively. 'Na' indicates the vehicle is already in an unstable state and cannot turn in a constant radius.

Steering angle	Speed (km/h)	10.0	20.0	30.0	40.0	50.0	70.0	100.0	≥110.0
1.67°	r (m)	103.3	105.5	110.4	118.3	127.3	146.5	164.6	Na
	$\delta/\delta_{neutral}$	1.0	1.1	1.1	1.2	1.3	1.5	1.6	Na
5°	r (m)	34.4	35.4	37.0	38.8	39.5	Na	Na	Na
	$\delta/\delta_{neutral}$	1.0	1.1	1.1	1.2	1.2	Na	Na	Na
7°	r (m)	24.6	25.4	26.3	26.9	Na	Na	Na	Na
	$\delta/\delta_{neutral}$	1.0	1.1	1.1	1.2	Na	Na	Na	Na

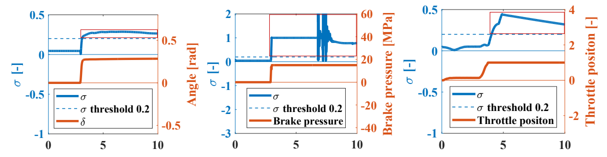
at low vehicle speed and a small steering angle, e.g., below 10km/h and 1.67°, the vehicle has a small slide slip, close to neutral steering, and hence $\delta/\delta_{neutral}$ is close to 1. Conversely, if $\delta/\delta_{neutral}$ is greater than a threshold, side slip happens. The simulation results shown in Table 2 indicate that the vehicle enters into an unstable state easily when $\delta/\delta_{neutral}$ is greater than 1.2, and thus we set the threshold of $\delta/\delta_{neutral}$ to 1.2 for the SD model by default. Due to the vehicle diversity and customized requirements, such as different types of vehicles or different operation conditions, the users can set a proper threshold when deploying SAID.

6.1.3 Accelerating/braking-based Anomaly Detection: We first demonstrate the impacts of threshold settings on abnormal detection performance. Since abnormal accelerating/braking motions can cause severe anomalies, we evaluate whether SAID can detect them by leveraging the BD models (cf. §4.3). Particularly, we simulate the side tire slips due to various injected messages related to throttle, brake pedal, and steering wheel angle.

We demonstrate the impacts of threshold settings of BD on the defense performances with various brake distance tests. The brake distance test is a simple but representative test to evaluate braking stability. In this experiment, we simulated the brake distances required under different road friction coefficients μ and slip ratio σ threshold settings (i.e., 0.05, 0.1, 0.2, 0.3, 0.5, 0.7, and 1.0) when the abnormal messages are injected to maximum brake force. The results are summarized in Table 3. Note that, when the slip ratio is 1.0, the tire is locked. From the simulation results, we can find, for every μ setting, the required brake distance first decreases with the σ threshold setting and then increases with it. The results also demonstrate that the shortest brake distance is required when

Table 3: The brake distances (m) of various parameters settings (i.e., the road friction coefficients μ and the thresholds of σ) for the accelerating/braking model.

Deceleration (km/h)	μ	Thresholds						
		$\sigma = 0.05$	$\sigma = 0.1$	$\sigma = 0.2$	$\sigma = 0.3$	$\sigma = 0.5$	$\sigma = 0.7$	Brake lock
100 - 0	0.9	99.3	76.6	40.4	42.3	46.8	48.1	52.8
	0.6	110.2	89.5	61.8	70.2	72.2	73.9	77.8
	0.3	156.3	142.7	141.8	146.8	150.0	151.0	151.8
80 - 0	0.9	62.7	52.2	26.1	27.1	30.0	30.8	32.8
	0.6	68.2	59.6	40.5	45.1	46.5	47.6	49.1
	0.3	99.0	93.9	92.2	94.9	97.0	97.6	97.6
60 - 0	0.9	35.1	32.3	15.2	15.3	16.8	17.3	19.4
	0.6	38.2	36.3	22.6	25.4	26.2	26.8	28.6
	0.3	54.9	54.1	52.0	53.8	55.0	55.3	55.8
40 - 0	0.9	16.0	16.0	7.1	7.0	7.4	7.7	8.8
	0.6	17.0	17.0	10.4	11.3	11.7	12.0	12.9
	0.3	24.7	24.8	23.4	24.2	24.6	24.8	25
20 - 0	0.9	4.1	4.2	2.2	2.0	1.9	1.9	2.3
	0.6	4.4	4.4	2.9	2.9	2.9	3.0	3.4
	0.3	6.6	6.5	6.0	6.1	6.2	6.2	6.4



(a) Steering Angle (b) Brake (c) Throttle

Figure 9: The perception of side tire slip due to the injected messages that change steering angle, throttle, and brake (the initial vehicle speed for (a) (b) and (c) are 160, 80 and 80 km/h). The red boxes indicate severe abnormal side tire slip motions.

the σ threshold is set to around 0.2, which corresponds to the general cases that the maximum friction generated when the slip ratio is between 0.1 and 0.2 [35, 37, 49]. Consequently, we also set the threshold of the slip ratio σ to 0.2 by default.

Fig. 9 shows the changes of the slip ratio σ when MIAs are launched through injecting messages to change the steering angle, brake pressure, and throttle position, respectively. The red boxes indicate severe abnormal slide tire slips occur. As shown in Fig. 9(b), when the tire is held still and locked completely, σ equals 1, obviously exceeding the threshold. Hence, based on the BD model in §4.3, SAID can correctly and timely capture the abnormal tire slip states when such MIAs are launched, indicating SAID *can effectively detect the MIAs launched to cause abnormal dynamic states based on the dynamics-based models*.

Answer to RQ1: By leveraging the vehicle dynamics models, SAID can correctly and timely identify the abnormal vehicle states, including roll, yaw, and accelerating/braking, before severe anomalies (i.e., rollover, abnormal steering angle, and side tire slip).

6.2 State Attack Defense

In this section, we use simulations, testbed, robotic cars, and real vehicles to evaluate whether SAID can identify the state layer injection attacks based on both the dynamics-based and rule-based policies presented in §5.3.

Table 4: Result of estimating rollover likelihood of NanoCar using LTR and defense against MIAs launched by increasing V_x.

V_x / Defense / δ	0.2 m/s		0.4 m/s		0.6 m/s		0.8 m/s		1 m/s	
	No	Yes	No	Yes	No	Yes	No	Yes	No	Yes
5.7°	0.02	0.02	0.04	0.04	0.17	0.17	0.30	0.30	0.48	0.48
11.5°	0.04	0.04	0.15	0.15	0.35	0.35	0.64	0.60	1.01	0.61
17°	0.06	0.06	0.23	0.22	0.52	0.51	0.96	0.60	1.31	0.59

Table 5: Result of estimating rollover likelihood of NanoCar using LTR and defense against MIAs launched by increasing δ.

δ / Defense / V_x	5.7°		8.6°		11.5°		14.3°		17°	
	No	Yes	No	Yes	No	Yes	No	Yes	No	Yes
0.2 m/s	0.02	0.02	0.03	0.03	0.04	0.04	0.05	0.05	0.06	0.06
0.6 m/s	0.17	0.17	0.26	0.26	0.35	0.35	0.43	0.43	0.52	0.52
1 m/s	0.48	0.49	0.76	0.64	0.97	0.61	1.11	0.60	1.21	0.62

6.2.1 *Dynamics-based Defense:*

We evaluate the dynamics-based defense through both simulations and robotic cars.

Simulation: When the dynamics-based defense is enabled, the simulation results of the rollover, under/oversteer, and slide tire slip show all the abnormal dynamic states detected in §6.1 are prevented.

Robotic Cars: We also applied the dynamics-based defense to various robotic cars for evaluation.

To evaluate the RD-based defense, we conduct experiments by injecting specific messages (i.e., control commands) into the three robotic cars to continuously increase the steering angle or the speed, which will lead to a high risk of rollover. Table 4 and 5 show the defense results when continuous messages/commands are injected into NanoCar to increase its speed V_x and steering angle δ, and the results include both the LTRs measured when defense function is disabled (i.e., "No") and enabled ("Yes"). For example, if an adversary launches a rollover attack by injecting messages to increase the car's speed to 1 m/s when the robotic car runs at the constant steering angle 17°, the RD-based defense can effectively detect such an attack when LTR researches the threshold (i.e., 0.6) and prevent the rollover accidents by discarding the injected commands and preventing LTR from exceeding the threshold, i.e., LTR equals 0.59.

The experiments on JetRacer and Scout Mini achieve similar results, and we just present the detailed evaluation results of NanoCar due to the page limitation. Consequently, the evaluation results of robotic cars show our RD-based defense can effectively and timely identify the injected abnormal messages and further discard them to keep the vehicle safe.

We also evaluate the SD-based defense by injecting messages into the robotic cars to continuously increase the steering angles or speed, which can cause abnormal under/oversteering dynamics during cornering. When SD-based defense is enabled, SAID defends against such attacks according to $^{\delta}/_{\delta_{neutral}}$, of which the larger values indicate the higher risks of slide slip. Table 6 and 7 show the results of defending against the attacks on NanoCar, which are launched by injecting messages to increase the speed V_x and the steering angler δ, respectively. From the results, we can find if the de-

Table 6: Result of estimating yaw stability of NanoCar using $^{\delta}/_{\delta_{neutral}}$ against the MIAs launched by increasing V_x.

V_x	0.2 m/s		0.4 m/s		0.6 m/s		0.8 m/s		1 m/s	
Defense / δ	No	Yes	No	Yes	No	Yes	No	Yes	No	Yes
5.7°	0.99	1.02	1.01	1.00	0.99	1.00	0.99	0.98	0.98	0.99
11.5°	1.00	1.00	1.00	1.00	1.01	1.00	1.03	1.02	1.08	1.08
17°	1.04	1.05	1.05	1.05	1.04	1.04	1.07	1.07	1.30	1.20

Table 7: Result of estimating yaw stability of NanoCar using $^{\delta}/_{\delta_{neutral}}$ against the MIAs launched by increasing δ.

δ	5.7°		8.6°		11.5°		14.3°		17°	
Defense / V_x (m/s)	No	Yes	No	Yes	No	Yes	No	Yes	No	Yes
0.2 m/s	1.00	1.00	1.00	0.99	1.00	1.00	0.99	0.98	1.02	1.03
0.6 m/s	0.99	0.99	0.99	1.00	0.99	1.00	1.03	1.03	1.03	1.03
1 m/s	0.98	0.98	1.00	0.99	1.08	1.08	1.20	1.19	1.29	1.19

fense function is enabled (i.e., "Yes"), all abnormal dynamic states are prevented, i.e., no $^{\delta}/_{\delta_{neutral}}$ exceeds 1.2, which is threshold specified in this experiment, indicating the SD-based defense can effectively prevent side slip attacks. For example, when the adversary continuously increases the speed under a steering angle of 17°, NanoCar enters into a severe slide slip dynamics ($^{\delta}/_{\delta_{neutral}} = 1.30$) if SD-based defense is disabled, otherwise it is still in a safe state, namely $^{\delta}/_{\delta_{neutral}}$ is still around 1.20.

6.2.2 Rule-based Abnormal State Defense: In this experiment, we evaluate SAID in both testbed and real vehicles. For the evaluations in the testbed, we launch MIAs by injecting attack messages to trigger abnormal states and meanwhile replaying the real road data. For the evaluations in real vehicles, we inject various diagnostic messages to trigger abnormal behaviors/functionalities.

Testbed: We instruct the "adversary" (Fig. 6) to dynamically send diagnostic requests to change the driving states when replaying the captured in-vehicle and onboard context data. The replayed data include 33,293 diagnostic messages and 30,654 tuples of sensor data. Meanwhile, 3,329 diagnostic requests are issued to change the driving states (e.g., open and close windows) or ECU statuses (e.g., update firmware, write data, and clear fault code). For instance, we send diagnostic requests to turn off the turn signal during cornering and modify the ECU configurations during running. Thus abnormal vehicle states are caused.

We also adopt various strategies to modify the parameters of normal diagnostic requests to cause abnormal vehicle states, including bit transform, value amplification, etc. For instance, the vehicle speed of real road data is below 70 km/h, which is always normal, but it exceeds 120 km/h after amplification and thus will violate specific speed related rules during replay. The experimental results show that all diagnostic requests, which can cause abnormal driving states, are dropped by SAID, and 87 state layer defense policies are hit (118 in total). All dropped messages are manually verified based on the recorded traffic.

Real Vehicles: We also deploy SAID on the seven rental cars

Table 8: The FN and FP of the dynamics-based defenses.

Dynamics Model	**LTR**			**Understeer/oversteer**			**Slip**		
	Threshold	FN	FP	Threshold	FN	FP	Threshold	FN	FP
Rate under specific threshold	0.4	1.7%	0	1	0.17%	0	0.1	2.3%	0
	0.5	2.1%	0	1.1	1.1%	0	<u>0.2</u>	4.5	0
	<u>0.6</u>	4.6%	0	<u>1.2</u>	3.4%	0	0.3	7.1%	0
	0.7	10.4%	0	1.3	64.7%	0	0.4	13.8%	0

and use an OBD-II dongle to inject attack messages. Due to the safety consideration, all the target functionalities are not safety-sensitive, including door, window, light, mirror, wipers, and speaker related operations, and the cars keep static with handbrake during message injection. The results show that all the injected messages are identified by SAID timely, and, if we assume the car runs on a special road, SAID also discards all the injected messages that can cause abnormal states. For instance, if we assume the cars run on a highway, i.e., speed is 80 km/h, all the messages, injected to open windows and doors, are discarded. Therefore, the results indicate that SAID can be deployed to real vehicles to defend against the MIAs and prevent abnormal vehicle states.

6.2.3 Detection Rate: We evaluate the detection rate, including FN (False Negative) and FP (False Positives), of the dynamics-based defenses with various threshold configurations. From the results shown in Table 8 (the default thresholds are underlined), we can see that, for all models, the FN rises as the threshold increases, and meanwhile, the FNs under default thresholds (i.e., the underlined values) are all below 5%. Meanwhile, SAID has no FP because no injected normal control message causes the vehicle dynamic to exceed the thresholds. Note that the FN also depends on the message injection patterns, more messages injected before the vehicle dynamics exceeds the threshold will result in a higher FN rate. We also evaluate FP using the messages collected from the real vehicles, containing no abnormal messages, and the result shows SAID detects no FP.

Answer to RQ2: Based on our state estimation approaches and defense policies, SAID can effectively defend against the MIAs that can cause abnormal vehicle states.

6.3 Functional Attack Defense

We evaluate the defense capacity of SAID at *network* and *service* layers by injecting abnormal CAN frames and diagnostic messages, respectively, when replaying real road data.

6.3.1 Network Layer Defense: Since the OBD-II port connects to IVN directly or through the gateway, the adversary can attack IVN by injecting well-prepared CAN frames through the OBD-II port. In this experiment, we instruct the OBD-II dongle to launch four types of attacks against the emulated car (`VCar` in Fig. 6), including replay attack, bus-off attack, fuzzy attack, and DoS attack, and evaluate whether SAID can identify such attack frames correctly and efficiently. Meanwhile, we instruct `VCar` to record all frames received from the internal OBD-II port and then check whether the attack

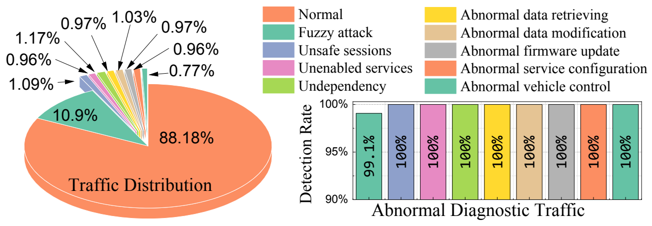


Figure 10: Distribution and detection rate of the launched diagnostic traffic (i.e., diagnostic messages).

frames are discarded by comparing the frames recorded with and without SAID deployed.

The experiment results show that all attack frames are filtered out by SAID at the network layer. More precisely, the frames injected for replay and bus-off attacks are dropped according to the CAN ID rules, because the CAN IDs belonging to [`0x0`, `700`) are only used by in-vehicle ECUs [3, 39, 90]. Thus, if the frames injected through the OBD-II port contain such CAN IDs, they are abnormal and dropped by SAID. During the fuzzy attack, 86.30% injected CAN frames are discarded according to the CAN IDs, and 11.87% injected frames are dropped due to the unmatched diagnostic sessions. Moreover, 1.83% of malicious CAN frames are filtered out due to the high transmission frequency. During the DoS attack, 87.16% and 12.84% of the injected frames are dropped according to their CAN IDs and frequency, respectively.

6.3.2 Service Layer Defense: To evaluate whether SAID can effectively identify the injected abnormal diagnostic messages according to the defense policies, we randomly inject diagnostic messages and change the contents of the replayed diagnostic messages on the testbed (i.e., Fig. 6). Meanwhile, we record the diagnostic messages on both `VCar` and Adversary sides. Then, we evaluate the defense performance of SAID by comparing the traffic recorded on both sides.

In this experiment, around one million diagnostic messages are sent to `VCar` with SAID, including 188,203 (18.82%) abnormal messages and 811,801 (81.18%) normal messages. The message distribution is shown on the left-hand side of Fig. 10, and the right-hand side of Fig. 10 shows the detection rates of the abnormal diagnostic messages. The experiment results show that apart from the fuzzy attack traffic, of which 99.1% are detected, all other types of attack traffic are detected. The fuzzy attacks trigger a larger number of negative diagnostic responses, and thus SAID detects such attacks according to the frequency of negative responses, which is under the specified threshold at the beginning of fuzzy attacks. Therefore, SAID does not identify the injected malicious messages at the beginning of fuzzy attacks, leading the false positives (0.9%). Such missed frames do not endanger the vehicle and will be discarded in IVN because only a few frames are injected and also cannot be processed by IVN.

We also compare SAID with the existing vehicle IDSs and the results are presented in Appendix-A.2.

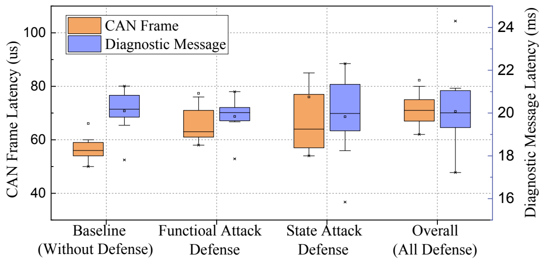


Figure 11: The overhead of SAID.

Answer to RQ3: The experimental results show that SAID can efficiently identify the abnormal driving states and prevent the MIAs from affecting the vehicles, thus outperforming existing IDSs.

6.4 Overhead

SAID inspects CAN frames and diagnostic messages at three layers, and meanwhile, additional latency will be incurred to both the CAN frames and diagnostic messages. To measure the incurred latency, we run SAID with CAN frame related rules, diagnostic message related rules, and overall rules enabled, respectively. Meanwhile, we measure the gaps between the timestamps when the frames and messages are captured at the external OBD-II port of SAID and the internal OBD-II port as the incurred latency.

Taking the latency incurred by SAID without any defense policy as the baseline, and then the other latency incurred with different configurations is shown in Fig. 11. Compared with the baseline, SAID incurs around 10 us to the CAN frames and negligible latency to the diagnostic messages. Note that the diagnostic messages can be divided into single frames and multiple frames. Because the latency of the messages based on a single frame is the same as that of the common CAN frames, we just show the latency of the messages depending on multiple CAN frames in this experiment.

Answer to RQ4: SAID just incurs around 10 us and negligible additional latency to the network and diagnostic data transmission respectively and is acceptable because the max bit speed of CAN is up to 1 Mbit/s [40].

7 Discussion and Future Work

In this paper, we propose SAID to defend against the MIAs launched from out-vehicle entities. That is, when an abnormal dynamic state is identified, SAID can either filter out the causing incoming messages or just warn the driver according to the causes of such a state. Particularly, if an incoming message from out-vehicle entities (e.g., OBD-II dongle) will let the vehicle enter abnormal maneuvers/states, SAID will reject it. However, it does not prevent any in-vehicle ECU/sensor from transmitting data and just warns the driver of the identified anomalies. Consequently, if a suspicious/unusual state is controlled by the driver, SAID just informs the driver.

Moreover, if the messages are gradually injected to cause anomalies not exceeding thresholds, the dynamics-based defense does not stop them because no dynamics-based defense is validated, but they can be detected by other defense policies (e.g., network/rule-based defense). In this paper, we propose to apply the dynamics-based defense to prevent injected messages from putting the vehicle in danger, and such defense can be regarded as the last line to protect vehicles from physical crashes when abnormal data is injected. In the future, we will extend SAID to prevent the anomalies caused by in-vehicle messages and provide defense suggestions.

Currently, we just leverage the explicit and common relationship among the data collected from different layers for attack defense. In future work, we will mine the implicit and vehicle-specific relationships between the vehicle states and the network/service layer traffic for defense. Also, SAID mainly focuses on the known vehicle states and motions currently, and we will leverage machine learning to detect unknown anomalies by combining multiple-layer information.

In addition, in this paper, we assume both software and hardware of SAID are reliable. Due to the centralized nature of SAID, we will use hardware security components and visualization techniques to improve its security in practice.

8 Related Work

Vehicular State-Aware Defense: The statistical features of CAN traffic have been used to detect abnormal IVN behaviors [64, 70–72]. For example, Müter et al. used the patterns (e.g., frequency, consistency, and correlation) of CAN messages to detect abnormal behaviors [71]. Then, an entropy-based approach was proposed to detect intrusions by measuring the entropy of CAN messages [70]. Miller et al. detected the suspicious IVN behaviors based on the distributions of intervals between CAN messages [64]. Narayanan et al. applied the Hidden Markov Model to predict abnormal IVN states based on specific in-vehicle information [72].

Physical State-Aware Defense: The physical control invariant has been used to detect vehicular misbehaviors [31, 32, 79]. For instance, Cho et al. proposed to detect vehicle misbehaviors by comparing the real-time brake data with a norm braking control model [31]. Afterward, Choi et al. applied the control-invariant models to detecting the attacks against robotic vehicles [32]. Then the framework *SAVIOR* leveraged robust physical invariant to detecting and preventing signal injection attacks [79]. Although SAID detects anomalies by using models different from these works, its dynamics-based defense also relies on the robust physical/dynamic invariant.

Machine Learning based Defense: The machine learning models have been used to detect the injected IVN data. Olufowobi et al. applied the adaptive cumulative sum algorithm to detecting statistical changes and intrusions in CAN bus message stream based on change-point detection techniques [77]. Moreover, Olufowobi et al. also presented a specification-based IDS using the response time analysis of CAN [78]. Taylor et al. used a Long Short-Term Memory neural network to predict the next data word originating from each sender on the bus and detect anomalies if a large deviation is found [87]. Kang et al. employed a deep neural network (DNN) to discriminate normal and hacking packets [51]. The Hidden Markov Model is also used to predict the abnormal vehicle states [73].

Timing-based Defense: Since each ECU has its specific timing features, researchers use such unique features to detect abnormal CAN frames. A clock-based intrusion detection system, named CIDS, is proposed to detect the abnormal CAN frames. It extracts the clock skews from the frames as the fingerprints of the ECUs and then models their clock behaviors [29]. To prevent the replay attack, Choi et al. presented a method to identify ECUs using inimitable characteristics of signals emitted from different ECUs [33]. To detect abnormal frames at the physical layer, Xu et al. proposed a reinforcement learning-based physical authentication scheme, which checks CAN frames according to voltage patterns [98].

Comparison: Table 11 summarizes the major differences between SAID and the existing vehicular IDSs from various aspects. First, the existing IDSs detect anomalies using only one type of vehicular data, such as sensor, CAN frame, or OBD-II message, and SAID detects attacks in multiple layers (details in §3.2) using four types of vehicular data for defense; Moreover, compared with the existing control/physical invariant based anomaly detection approaches, SAID detects MIAs leveraging three VD models along with the RB (Rule-based) models; Since more types of data are utilized by SAID to detect MIAs, it can detect more types of attacks than the existing IDSs; It is worth noting that the Choi et al. [32] and Raul et al. [79] focus on detecting attacks on the robotic vehicles leveraging the control/physical invariants. Although SAID detects abnormal vehicle states using the specific vehicle dynamics models, which are different from the laws used by them, the dynamics-based defense capacity of SAID also depends on the control/physical invariant of the vehicles. In addition, SAID does not require to refit the vehicles for deployment and also has the capacity of preventing attacks; Also, we evaluate SAID with different experiment environments, including real cars, prototype systems, and simulations.

9 Conclusion

In this paper, we propose SAID, a new state-aware defense system against vehicular MIAs. It employs the vehicle dynamics models to estimate the vehicle states with the context information from both IVN and onboard sensors for defending against the attacks aiming to cause abnormal vehicle motions. Moreover, it exploits the eco/safe driving requirements and auto insurance policies to build rule-based defense policies for defending against more anomalies. SAID adopts a cross-layer architecture and can detect abnormal CAN frames and diagnostic messages at the network layer and service layer. The various types of evaluation results show SAID can effectively and efficiently detect various MIAs based on the rule-based and dynamics-based defense policies.

Acknowledgment

We sincerely thank Prof. Alvaro Cardenas for shepherding our paper and the anonymous reviewers for their constructive comments. This work is partly supported by Hong Kong ITF Project (No. ITS/197/17FP), Hong Kong RGC Project (No. PolyU15223918), NSFC (No. 62002306, 61872438), HKPolyU Start-up Fund (ZVU7), CCF-Tencent Open Research Fund (ZDCK), the Fundamental Research Funds for the Central Universities (Zhejiang University NGICS Platform ZJUNGICS2021017, K20200019), Leading Innovative and Entrepreneur Team Introduction Program of Zhejiang (No. 2018R01005), as well as the National Science Foundation (NSF) under Grant No. 1700544. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF.

References

[1] Road vehicles–diagnostics communication over controller area networks (docan)–part 2: Transport protocol and network layer services. 2011.

[2] Road vehicles–diagnostics communication over controller area networks (docan)–part 3: Implementation of unified diagnostic services (uds on can). 2011.

[3] Road vehicles–diagnostics communication over controller area networks (docan)–part 4: Requirements for emissions-related systems. 2011.

[4] Road vehicles–unified diagnostic services (uds)–part 3: Unified diagnostic services on can implementation (udsoncan), 2012.

[5] Road vehicles–unified diagnostic services (uds)–part 1: Specification and requirements, 2013.

[6] Road vehicles–unified diagnostic services (uds)–part 2: Session layer services, 2013.

[7] Road vehicles–communication between vehicle and external equipment for emissions-related diagnostics–part 5: Emissions-related diagnostic services, 2015.

[8] Birmingham personal injury: How can i avoid accidents due to oversteer and understeer? https://www.morrisbart.com/blog/avoid-accidents-oversteer-understeer-birmingham/, 2016.

[9] Defining Vehicle Dynamics. https://policedriver.com/defining-vehicle-dynamics, 2017.

[10] Safe driving tips. https://www.td.gov.hk/mini_site/2017sdhc/eng/sdtips.html, 2017.

[11] Eco driving. https://www.citroen.com.hk/citroen-universe/environment/eco-driving/, 2019.

[12] Eco driving tips. https://www.nidirect.gov.uk/articles/eco-safe-driving-cars-lorries-and-buses, 2019.

[13] Ecodriving. https://www.cieca.eu/sites/default/files/documents/projects_and_studies/ECOWILL_FINAL_REPORT.pdf, 2019.

[14] Hackers remotely kill a jeep on the highway. https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/, 2019.

[15] Keen lab hackers managed to take control of tesla vehicles again. https://electrek.co/2017/07/28/tesla-hack-keen-lab/, 2019.

[16] Safe driving. https://www.valeofglamorgan.gov.uk/en/living/Roads/road_safety/Safe-Driving.aspx, 2019.

[17] Safe driving. https://www.betterhealth.vic.gov.au/health/HealthyLiving/Safe-driving, 2019.

[18] Safe driving tips. https://www.in.gov/indot/2360.htm, 2019.

[19] Masato Abe. *Vehicle handling dynamics: theory and application*. Butterworth-Heinemann, 2015.

[20] ARGUS. A remote attack on the bosch drivelog connector dongle, 2017.

[21] Omid Avatefipour and Hafiz Malik. State-of-the-art survey on in-vehicle network communication (canbus) security and vulnerabilities. *arXiv preprint arXiv:1802.01725*, 2018.

[22] 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 *Proc. ACM CCS*, 2019.

[23] Paul Carsten, Todd R Andel, Mark Yampolskiy, and Jeffrey T McDonald. In-vehicle networks: Attacks, vulnerabilities, and proposed solutions. In *Proc. ACISRC*, 2015.

[24] Mario Casillo, Simone Coppola, Massimo De Santo, Francesco Pascale, and Emanuele Santonicola. Embedded intrusion detection system for detecting attacks

over can-bus. In *Proc. IEEE ICSRS*, 2019.

[25] Teck Kai Chan, Cheng Siong Chin, Hao Chen, and Xionghu Zhong. A comprehensive review of driver behavior analysis utilizing smartphones. *IEEE TITS*, 2019.

[26] Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, Stefan Savage, Karl Koscher, Alexei Czeskis, Franziska Roesner, Tadayoshi Kohno, et al. Comprehensive experimental analyses of automotive attack surfaces. In *Proc. USENIX Security*, 2011.

[27] Dongyao Chen, Kyong-Tak Cho, Sihui Han, Zhizhuo Jin, and Kang G Shin. Invisible sensing of vehicle steering with smartphones. In *Proc. ACM MobiSys*, 2015.

[28] Kyong-Tak Cho and Kang G Shin. Error handling of in-vehicle networks makes them vulnerable. In *Proc. ACM CCS*, 2016.

[29] Kyong-Tak Cho and Kang G Shin. Fingerprinting electronic control units for vehicle intrusion detection. In *Proc. USENIX Security*, 2016.

[30] Kyong-Tak Cho and Kang G Shin. Viden: Attacker identification on in-vehicle networks. In *Proc. ACM CCS*, 2017.

[31] Kyong-Tak Cho, Kang G Shin, and Taejoon Park. Cps approach to checking norm operation of a brake-by-wire system. In *Proc. ACM/IEEE ICCPS*, 2015.

[32] Hongjun Choi, Wen-Chuan Lee, Yousra Aafer, Fan Fei, Zhan Tu, Xiangyu Zhang, Dongyan Xu, and Xinyan Deng. Detecting attacks against robotic vehicles: A control invariant approach. In *Proceedings of the 2018 ACM SIGSAC CCS*, 2018.

[33] Wonsuk Choi, Hyo Jin Jo, Samuel Woo, Ji Young Chun, Jooyoung Park, and Dong Hoon Lee. Identifying ecus using inimitable characteristics of signals in controller area networks. *IEEE TVT*, 67(6):4757–4770, 2018.

[34] Wonsuk Choi, Kyungho Joo, Hyo Jin Jo, Moon Chan Park, and Dong Hoon Lee. Voltageids: Low-level communication characteristics for automotive intrusion detection system. *IEEE TIFS*, 13(8), 2018.

[35] Matteo Corno, Mathieu Gerard, Michel Verhaegen, and Edward Holweg. Hybrid abs control using force measurement. *IEEE TCST*, 20(5):1223–1235, 2011.

[36] Moustapha Doumiati, Aa Victorino, Ali Charara, and Daniel Lechner. Lateral load transfer and normal forces estimation for vehicle safety: experimental test. *Vehicle System Dynamics*, 47(12):1511–1533, 2009.

[37] Sergey Drakunov, Umit Ozguner, Peter Dix, and Behrouz Ashrafi. Abs control using optimum search via sliding modes. *IEEE TCST*, 3(1):79–85, 1995.

[38] Andrew Evans. Oversteer and understeer explained. https://www.carwow.co.uk/guides/glossary/what-is-oversteer-understeer, 2015.

[39] Eric Evenchick. Hopping on the can bus-automotive security and the canard toolkit. *Black Hat Asia*, 2015.

[40] International Organization for Standardization. Road vehicles–interchange of digital information–controller area network (can) for high-speed communication. *ISO/DIS Standard 11898:1993*, 1993.

[41] Arun Ganesan, Jayanthi Rao, and Kang Shin. Exploiting consistency among heterogeneous sensors for vehicle anomaly detection. Technical report, SAE Technical Paper, 2017.

[42] Mabrouka Gmiden, Mohamed Hedi Gmiden, and Hafedh Trabelsi. An intrusion detection method for securing in-vehicle can bus. In *Proc. IEEE STA*, 2016.

[43] Bogdan Groza and Pal-Stefan Murvay. Efficient intrusion detection with bloom filtering in controller area networks. *IEEE TIFS*, 14(4), 2018.

[44] Rudolf Hackenberg, Nils Weiss, Sebastian Renner, and Enrico Pozzobon. Extending vehicle attack surface through smart devices, 2017.

[45] Shawn Hartzell and Christopher Stubel. Automobile can bus network security and vulnerabilities, 2017.

[46] HCRL. Can intrusion dataset. https://sites.google.com/a/hksecurity.net/ocslab/Datasets/%7B/can%7D-intrusion-dataset, 2021.

[47] Bing He, Xiaolin Chen, Dian Zhang, Siyuan Liu, Dawei Han, and Lionel M Ni. Pbe: Driver behavior assessment beyond trajectory profiling. In *Proc. EMLL-PKDD*, 2018.

[48] Computer Hope. Electronic Control Unit. https://www.computerhope.com/jargon/e/ecu.htm, 2017.

[49] Bengt Jacobson. Vehicle dynamics compendium. Technical report, Chalmers University of Technology, 2019.

[50] Pengfei Jing, Qiyi Tang, Yuefeng Du, Lei Xue, Xiapu Luo, Ting Wang, Sen Nie, and Shi Wu. Too good to be safe: Tricking lane detection in autonomous driving with crafted perturbations. In *Proc. USENIX Security*, 2021.

[51] Min-Joo Kang and Je-Won Kang. Intrusion detection system using deep neural network for in-vehicle net-

work security. *PloS One*, 11(6), 2016.

[52] Satya Katragadda, Paul J Darby, Andrew Roche, and Raju Gottumukkala. Detecting low-rate replay-based injection attacks on in-vehicle networks. *IEEE Access*, 8:54979–54993, 2020.

[53] Gregor Klancar, Andrej Zdesar, Saso Blazic, and Igor Skrjanc. *Wheeled mobile robotics: from fundamentals towards autonomous systems*. Butterworth-Heinemann, 2017.

[54] Dan Klinedinst and Christopher King. On board diagnostics: Risks and vulnerabilities of the connected vehicle. *CERT Coordination Center, Tech. Rep*, 2016.

[55] Konstantin Korishchenko, Ivan Stankevich, Nikolay Pilnik, and Daria Marchenko. Usage-based vehicle insurance: Driving style factors of accident probability and severity. *arXiv preprint arXiv:1910.00460*, 2019.

[56] Chad Larish, Damrongrit Piyabongkarn, Vasilios Tsourapas, and Rajesh Rajamani. A new predictive lateral load transfer ratio for rollover prevention systems. *IEEE TVT*, 2013.

[57] Kiho Lim and D Manivannan. An efficient protocol for authenticated and secure message delivery in vehicular ad hoc networks. *Vehicular Communications*, 4:30–37, 2016.

[58] Dawei Lyu, Lei Xue, Yu Le, and Xiapu Luo. Remote attacks on vehicles by exploiting vulnerable telematics. https://www.yumpu.com/en/document/read/56559249/remote-attacks-on-vehicles-by-exploiting-vulnerable-telematics, 2016.

[59] Yu-Luen Ma, Xiaoyu Zhu, Xianbiao Hu, and Yi-Chang Chiu. The use of context-sensitive insurance telematics data in auto insurance rate making. *Transportation Research Part A: Policy and Practice*, 113:243–258, 2018.

[60] Mirco Marchetti and Dario Stabili. Anomaly detection of can bus messages through analysis of id sequences. In *Proc. IEEE IV*, 2017.

[61] Mirco Marchetti, Dario Stabili, Alessandro Guido, and Michele Colajanni. Evaluation of anomaly detection for in-vehicle networks through information-theoretic algorithms. In *Proc. IEEE RTSI*, 2016.

[62] Fabio Martinelli, Francesco Mercaldo, Vittoria Nardone, and Antonella Santone. Car hacking identification through fuzzy logic algorithms. In *Proc. IEEE FUZZ*, 2017.

[63] Mechanical Simulation. Carsim. https://www.carsim.com, 2020.

[64] Charlie Miller and Chris Valasek. Adventures in automotive networks and control units. *Def Con*, 2013.

[65] Charlie Miller and Chris Valasek. Ioactive adventures in automotive networks and control units. *Defcon*, 2014.

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

[67] Michael R Moore, Robert A Bridges, Frank L Combs, Michael S Starr, and Stacy J Prowell. Modeling inter-signal arrival times for accurate detection of can bus signal injection attacks: a data-driven approach to in-vehicle intrusion detection. In *Proc. ACM CISRC*, 2017.

[68] Mohamed Moussa, Adel Moussa, and Naser El-Sheimy. Steering angle assisted vehicular navigation using portable devices in gnss-denied environments. *Sensors*, 19(7), 2019.

[69] Philipp Mundhenk, Sebastian Steinhorst, Martin Lukasiewycz, Suhaib A Fahmy, and Samarjit Chakraborty. Lightweight authentication for secure automotive networks. In *Proc. DATE*, 2015.

[70] Michael Müter and Naim Asaj. Entropy-based anomaly detection for in-vehicle networks. In *Proc. IEEE IV*, 2011.

[71] Michael Müter, André Groll, and Felix C Freiling. A structured approach to anomaly detection for in-vehicle networks. In *Proc. IEEE IAS*, 2010.

[72] Sandeep Nair Narayanan, Sudip Mittal, and Anupam Joshi. Using data analytics to detect anomalous states in vehicles. *arXiv preprint arXiv:1512.08048*, 2015.

[73] Sandeep Nair Narayanan, Sudip Mittal, and Anupam Joshi. Obd_securealert: An anomaly detection system for vehicles. In *Proc. IEEE SMARTCOMP*, 2016.

[74] Thomas Nolte, Hans Hansson, and Lucia Lo Bello. Automotive communications-past, current and future. In *Proc. IEEE ETFA*, 2005.

[75] Nasser Nowdehi, Wissam Aoudi, Magnus Almgren, and Tomas Olovsson. Casad: Can-aware stealthy-attack detection for in-vehicle networks. *arXiv preprint arXiv:1909.08407*, 2019.

[76] Shuji Ohira, Araya Kibrom Desta, Ismail Arai, Hiroyuki Inoue, and Kazutoshi Fujikawa. Normal and malicious sliding windows similarity analysis method for fast and accurate ids against dos attacks on in-vehicle networks. *IEEE Access*, 8:42422–42435, 2020.

[77] Habeeb Olufowobi, Uchenna Ezeobi, Eric Muhati, Gay-

lon Robinson, Clinton Young, Joseph Zambreno, and Gedare Bloom. Anomaly detection approach using adaptive cumulative sum algorithm for controller area network. In *Proc. ACM AutoSec*, 2019.

[78] Habeeb Olufowobi, Clinton Young, Joseph Zambreno, and Gedare Bloom. Saiducant: Specification-based automotive intrusion detection using controller area network (can) timing. *IEEE TVT*, 69(2):1484–1494, 2019.

[79] Raul Quinonez, Jairo Giraldo, Luis Salazar, Erick Bauman, Alvaro Cardenas, and Zhiqiang Lin. Savior: Securing autonomous vehicles with robust physical invariants. In *Proc. USENIX Security*, 2020.

[80] Andreea-Ina Radu and Flavio D Garcia. Leia: A lightweight authentication protocol for can. In *Proc. ESORCS*, 2016.

[81] Rajesh Rajamani. *Vehicle dynamics and control*. New York: Springer, 2012.

[82] Raimondo Sferco, Yves Page, Jean-Yves Le Coz, and Paul A Fay. Potential effectiveness of electronic stability programs (esp)-what european field studies tell us. Technical report, SAE Technical Paper, 2001.

[83] Xiupeng Shi. Accident risk assessment and prediction using surrogate indicators and machine learning. 2019.

[84] Yasser Shoukry, Paul Martin, Paulo Tabuada, and Mani Srivastava. Non-invasive spoofing attacks for anti-lock braking systems. In *Proc. CHES*, 2013.

[85] Dario Stabili, Mirco Marchetti, and Michele Colajanni. Detecting attacks to internal vehicle networks through hamming distance. In *Proc. AEIT*, 2017.

[86] Sara Stančin and Sašo Tomažič. Time-and computation-efficient calibration of mems 3d accelerometers and gyroscopes. *Sensors*, 14(8):14885–14915, 2014.

[87] Adrian Taylor, Sylvain Leblanc, and Nathalie Japkowicz. Anomaly detection in automobile control network data with long short-term memory networks. In *Proc. IEEE DSAA*, 2016.

[88] Andrew Tomlinson, Jeremy Bryans, Siraj Ahmed Shaikh, and Harsha Kumara Kalutarage. Detection of automotive can cyber-attacks by identifying packet timing anomalies in time windows. In *Proc IEEE DSN-W*, 2018.

[89] Jo Van Bulck, Jan Tobias Mühlberg, and Frank Piessens. Vulcan: Efficient component authentication and software isolation for automotive control networks. In *Proc. ACM ACSAC*, 2017.

[90] Jan Van den Herrewegen and Flavio D Garcia. Beneath the bonnet: a breakdown of diagnostic security. In *Proc. ESORICS*, 2018.

[91] Anton T Van Zanten et al. Evolution of electronic control systems for improving the vehicle dynamic behavior. In *Proc. AVEC*, 2002.

[92] Qiyan Wang and Sanjay Sawhney. Vecure: A practical security framework to protect the can bus of vehicles. In *Proc. IEEE IOT*, 2014.

[93] Armin Wasicek, Mert D Pese, André Weimerskirch, Yelizaveta Burakova, and Karan Singh. Context-aware intrusion detection in automotive control systems. In *ESCAR USA Conference*, 2017.

[94] Zhuo Wei, Yanjiang Yang, and Tieyan Li. Authenticated can communications using standardized cryptographic techniques. In *Proc. ISPEC*, 2016.

[95] Greg Welch, Gary Bishop, et al. An introduction to the kalman filter. 1995.

[96] Haohuang Wen, Qi Alfred Chen, and Zhiqiang Lin. Plug-n-pwned: Comprehensive vulnerability analysis of obd-ii dongles as a new over-the-air attack surface in automotive iot. In *Proc. USENIX Security*, 2020.

[97] 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 TITS*, 2015.

[98] Tangwei Xu, Xiaozhen Lu, Liang Xiao, Yuliang Tang, and Huaiyu Dai. Voltage based authentication for controller area networks with reinforcement learning. In *Proc. IEEE ICC*, 2019.

[99] Shaopu Yang, Yongjie Lu, and Shaohua Li. An overview on vehicle dynamics. *International Journal of Dynamics and Control*, 1(4), 2013.

[100] Yun Yang, Zongtao Duan, and Mark Tehranipoor. Identify a spoofing attack on an in-vehicle can bus based on the deep features of an ecu fingerprint signal. *Smart Cities*, 2020.

[101] Le Yu, Yangyang Liu, Pengfei Jing, Xiapu Luo, Lei Xue, Kaifa Zhao, Yajin Zhou, Ting Wang, Guofei Gu, Sen Nie, and Shi Wu. Towards automatically reverse engineering vehicle diagnostic protocols. In *Proc. USENIX Security*, 2022.

[102] Yu Zhang, Binbin Ge, Xiang Li, Bin Shi, and Bo Li. Controlling a car through obd injection. In *Proc. CSCloud*, 2016.

A Appendix

A.1 Defense Results of Simulations

We evaluate the defense performances of SAID when the MIAs are launched by injecting various types of messages simultaneously. As shown in Table 9, we simulate 22 MIAs that are launched under various initial vehicle states and through injecting different messages. To evaluate whether SAID can defend against these MIAs, we use SAID to protect the vehicle from entering abnormal motions. The right column of Table 9 presents the defense results and all abnormal motions are prevented after SAID is applied.

A.2 Comparison with Exiting Vehicular IDS

We compare SAID with existing vehicular IDSs using the public CAN bus dataset [46], and the results are shown in Table 10. Note that existing IDSs do not consider the server layer and state layer attacks, and their dataset just contains the network layer attack traffic, including replay attack (a.k.a., spoof attack and fabricate attack), DoS attack, and fuzzy attack. The results show that apart from SAID none of the existing IDS can identify all these injected malicious messages, because only SAID considers context information. More precisely, exiting IDSs are deployed in IVN and detect anomalies with the stability features of IVN traffic, such as intervals and sequences. Hence, they cannot identify the injected abnormal messages with normal traffic patterns. Also, the normal diagnostics may cause false positives due to unusual IVN traffic patterns produced. In contrast, SAID is deployed between the IVN and out-vehicle entities and detects anomalies considering both the traffic features and the context information. Therefore, when such abnormal diagnostic traffic is injected, SAID can timely identify them according to their message IDs, which can be only used by IVN messages.

Table 9: The defense results of various MIAs launched at high and middle speed (i.e., 90 km/h and 60 km/h) as well as various steering wheel angles (δ_w). ✓ indicates SAID successfully detected and prevented the abnormal vehicle motions.

Attack Purpose	Initial state		MIAs (Injected Messages)			Defense Result
	Velocity	Steering wheel	Throttle	Brake	Steering wheel	
Rollover	60 km/h	25°	increase	Na	increase	✓
	60 km/h	25°	Na	increase	increase	✓
	60 km/h	15°	increase	Na	increase	✓
	60 km/h	15°	Na	increase	increase	✓
	90 km/h	25°	increase	Na	increase	✓
	90 km/h	25°	Na	increase	increase	✓
	90 km/h	15°	increase	Na	increase	✓
	90 km/h	15°	Na	increase	increase	✓
Under/ Oversteer	60 km/h	25°	increase	Na	increase	✓
	60 km/h	25°	Na	increase	increase	✓
	60 km/h	15°	increase	Na	increase	✓
	60 km/h	15°	Na	increase	increase	✓
	90 km/h	15°	increase	Na	increase	✓
	90 km/h	15°	Na	increase	increase	✓
Side slip	60 km/h	25°	increase	Na	increase	✓
	60 km/h	25°	Na	increase	increase	✓
	60 km/h	15°	increase	Na	increase	✓
	60 km/h	15°	Na	increase	increase	✓
	90 km/h	25°	increase	Na	increase	✓
	90 km/h	25°	Na	increase	increase	✓
	90 km/h	15°	increase	Na	increase	✓
	90 km/h	15°	Na	increase	increase	✓

Table 10: Comparison between the existing vehicular IDSs (√ and × indicate the IDS has or has no such an ability).

IDS	Network Layer			Service Layer	State Layer
	DoS	Replay	Fuzzy		
Muter et al.[70]	×	0.9957	0.9996	×	×
	0.9685	0.9546	0.9647	×	×
Gmiden et al.[42]	×	0.9765	0.9912	×	×
Cho et al.[29]	0.4472	0.7063	0.3589	×	×
Kang et al.[51]	0.9647	0.9912	0.7517	×	×
Taylor et al.[87]	0.7818	0.9136	0.6365	×	×
Moore et al.[67]	0.9382	0.9663	0.9527	×	×
Marchetti et al.[60]	0.9616	0.8367	0.9819	×	×
Tomlinson et al.[88]	0.9714	0.8901	0.9744	×	×
	×	0.8462	0.8665	×	×
Stabili et al.[85]	0.9977	0.8481	0.952	×	×
Ohira et al.[76]	0.9924	0.8374	0.9508	×	×
SAID	1	1	1	√	√

Table 11: Comparison with the existing vehicular IDSs on the exploited data, the type of detection algorithms, the target attacks,the evaluation environments, the requirement of refitting, and the prevention capacity. RB, ML, PI, and VD indicate the rule-based, machine-learning based, physical invariant and vehicle dynamics based detection approach, respectively. It is worth noting that Choi et al. [32] and Raul et al. [79] focus on detecting attacks on the robotic vehicles.

IDS	Used data				Type	Targets	Evaluation environment	Refit	Prevention
	Sensor	CAN	OBD	UDS					
Muter et al. [70]	×	√	×	×	RB	DoS, Spoof	Real car	√	×
Gmiden et al. [42]	×	√	×	×	RB	DoS, Spoof	Simulation	√	×
Cho et al. [29]	×	√	×	×	RB	DoS, Spoof	Real car, Prototype, Simulation	√	×
Kang et al. [51]	×	√	×	×	ML (DNN)	Spoof	Simulation	√	×
Taylor et al. [87]	×	√	×	×	ML (RNN and LSTM)	Spoof	Simulation	√	×
Moore et al. [67]	×	√	×	×	RB	DoS, Spoof	Real car	√	×
Marchetti et al. [60]	×	√	×	×	RB	Spoof, Fuzzy	Real car	√	×
Tomlinson et al. [88]	×	√	×	×	RB	DoS, Spoof	Simulation	√	×
Stabili et al. [85]	×	√	×	×	RB	Spoof, Fuzzy	Real car, Simulation	√	×
Ohira et al. [76]	×	√	×	×	RB	DoS, Spoof, Fuzzy	Simulation	√	×
Casillo et al. [24]	×	√	×	×	ML (Bayesian)	Spoof	Simulation	√	×
Cho et al. [30]	×	√	×	×	RB	Spoof	Real car, Prototype	√	×
Ganesan et al. [41]	√	×	×	×	RB	Spoof	Simulation	√	×
Wasicek et al. [93]	×	×	√	×	ML (Bottleneck ANN)	Spoof	Real car	√	×
Narayanan et al. [73]	×	×	√	×	ML (HMM)	Spoof	Real car	√	×
Choi et al. [32]	√	×	×	×	PI	Signal Injection	Robotic Vehicles	√	√
Raul et al. [79]	√	×	×	×	PI	Signal Injection	Robotic Vehicles	√	√
SAID	√	√	√	√	RB and VD	DoS, Spoof, Bus-off, Fuzzy, OBD/UDS message injection	Real car, Prototype, Simulation	×	√