Journal of Hardware and Systems Security
https://doi.org/10.1007/s41635-024-00150-w

VISE: Digital Twin Exploration for Automotive Functional Safety and Cybersecurity

Md Rafiul Kabir[1] · **Sandip Ray**[1]

Received: 15 August 2023 / Accepted: 2 May 2024

Abstract
The automotive industry relies on modern automobile systems, which are complex distributed embedded systems. These systems consist of numerous electronic control units, sensors, and actuators, all interconnected through multiple in-vehicle networks. However, the integration of these diverse components can introduce subtle errors that may be exploited by malicious individuals, leading to severe consequences. To address this, we employ our developed digital twin platform that facilitates the exploration of automotive functional safety and cyber security. Within this environment, we implement safety and security scenarios that allow for interaction with various electronic control units, as well as the simulation of sensory inputs and actuation outputs. By analyzing several vehicular use case interactions, we address critical safety and security concerns through this digital twin and facilitate a comprehensive exploration solution. We also establish the foundation for security policies and countermeasures that can be freely explored within the digital twin environment.

Keywords Digital twin · Automotive · Electronics · Software process · Simulation · Safety · Security

1 Introduction

The ongoing digital transformation has brought about significant changes in embedded systems, particularly with the emergence of the Internet of Things (IoT) and advancements in artificial intelligence [1]. While these developments have introduced pervasive connectivity, they have also exposed us to increased risks of electronic failures and cyberattacks. To ensure our well-being, it is imperative to develop technologies that systematically explore the safety and security aspects of critical cyber-physical systems.

Automotive systems are quintessential cyber-physical systems that exemplify the critical need for robust safety and security measures. Modern automobiles are equipped with numerous electronic control units (ECUs), sensors, actuators, substantial software, and in-vehicle networks. These systems also possess interfaces for external communication. The integration of these components creates opportunities for compromising safety and security, potentially leading to catastrophic accidents. One major concern is the introduction of connectivity to components that were not originally designed with it in mind. For instance, ECUs were initially intended to receive commands from internal components and share information within the same CAN bus without authentication or validation; however, connectivity exposes them to exploitation by hackers. Similarly, sensors and actuators, which contribute to peripheral activities for ECU performance, are susceptible to failures or adversarial compromises.

In this paper, we develop a platform, VISE (for "**V**ehi**i**cular Safety and **S**ecurity **E**xploration"), for exploring the safety and security aspects of automotive systems. VISE is a digital twin infrastructure designed to investigate component failures and security compromises. It enables the exploration of interaction among various vehicular components while abstracting the low-level implementation details of ECUs, sensors, actuators, etc. We demonstrate VISE in representative platform-level use cases.

A previous paper [2] developed an initial vision for the safety and security of automotive systems through the systematic design of virtual prototyping solutions. This framework has been applied successfully to explore various automotive use cases. However, its scope was limited

✉ Md Rafiul Kabir
kabirm@ufl.edu

Sandip Ray
sandip@ece.ufl.edu

[1] Department of ECE, University of Florida, Gainesville, FL 32611, USA

Published online: 17 May 2024

to functionality exploration and optimization opportunities and lacked computational processes necessary for safety and security exploration. ViSE overcomes these deficiencies and enables comprehensive simulation of various safety and security scenarios. We are not aware of any other platform that enables such exploration.

The remainder of the paper is organized as follows. Section 2 discusses the background and relevant research in this area. Section 3 presents the system-level digital twin architecture of our platform. In Section 4, we discuss the use of ViSE for two illustrative vehicular use cases. We demonstrate our digital twin approach for functional safety and security compromise exploration in Sections 5 and 6, respectively. In Section 7, we discuss the exploration of countermeasures for safety and security. We conclude in Section 8.

2 Background and Related Work

2.1 Digital Twin

The concept of a digital twin (DT) has gained widespread popularity in the development of advanced cyber-physical systems, smart systems, and applications infused with IoT technology [3]. Initially, conceptualized by Grieves as the underlying paradigm for product lifecycle management [4], digital twins distinguish themselves from virtual platforms by focusing on the physical behavior of entities rather than abstracting software layers. It is defined as computer-based models or virtual prototypes that replicate, emulate, or mirror the existence of a physical entity, extending beyond mere simulations or virtual representations [5, 6]. By combining physical and virtual assets through the Internet of Things, digital twins provide numerous benefits such as simulation, prediction, and monitoring. As a result, various industries including manufacturing, connected and autonomous vehicles, healthcare, energy, and city planning are experiencing a revolutionary transformation through the adoption of DT technologies [7].

To achieve comprehensive improvement in every aspect of IoT systems, DTs can be constructed from physical entities that provide feedback based on simulation results. This approach facilitates IoT application development, encompassing exploration, optimization, and security aspects. As a widely adopted virtual prototyping technology, DT serves as a bridge between physical and virtual components, supporting *object* and *process twinning* [8]. Digital twinning of objects involves creating virtual counterparts with real-time automated monitoring, seamlessly integrating them into automatic cyber-physical systems. On the other hand, the digital twinning of a process entails virtualizing the entire CPS subsystem. This all-encompassing approach unlocks immense potential for enhancing IoT systems and driving efficiency and innovation in various industries.

2.2 Related Work

The utilization of digital twins in the IoT-based cyber-physical industry is an emerging technology. While research groups have extensively explored the domain of functional safety and cyber security, the application of digital twins in this context is relatively new. Almeaibed et al. [9] proposed to establish a universal framework for vehicular digital twins, aiming to streamline the processes of data collection, data processing, and analytics. In order to showcase the efficiency of this proposed approach, a case study was conducted on a vehicle follower model that involved manipulating radar sensor measurements to induce a collision. An IoT system also employed the concept of a digital-twin-driven safety environment to create virtual representations of vehicle status transitions for platform health monitoring [10]. Damjanovic-Behrendt [11] presented a digital twin demonstrator as a means to design and implement privacy enhancement mechanisms. This approach is utilized to identify privacy issues and mitigate potential breaches and associated risks faced by smart car drivers through connected infotainment applications and services. Shadrin et al. [12] analyzed regulatory documents pertaining to operational and safety assessments to explore the applicability of digital twin technology in monitoring the safety of highly automated vehicles. They prescribed procedures to create digital twins of highly automated vehicles presented with field and virtual tests. Safar et al. [13] proposed the integration of a virtual platform with the V-model of automotive software development, enabling streamlined verification and validation at the SoC, ECU, and system levels.

For domain-specific needs, several automotive simulators and security exploration platforms have emerged. SUMO [14] serves as an open-source miniature traffic simulation platform, while CARLA [15] provides a comprehensive framework for the development, training, and validation of autonomous driving systems. Another tool Metropolis [16] primarily focused on design space exploration of automotive platforms, emphasizing software distribution, architecture, and network configuration. In situations where real vehicles are unavailable for experimentation, Yang et al. [17] developed a digital twin prototype, enabling multi-vehicle experiments to be conducted virtually. For hands-on security exploration, AutoHaL [18] has proven to be an effective platform, specifically designed to investigate ranging sensor attacks. Scalas and Giacinto [19] proposed a systematic approach to consider crucial cybersecurity factors during the development of modern automobiles, aiming to consolidate knowledge and enhance security measures in the field. Additionally, Owoputi et al. [20] used an immersive virtual environment to investigate safety vulnerabilities in automotive systems

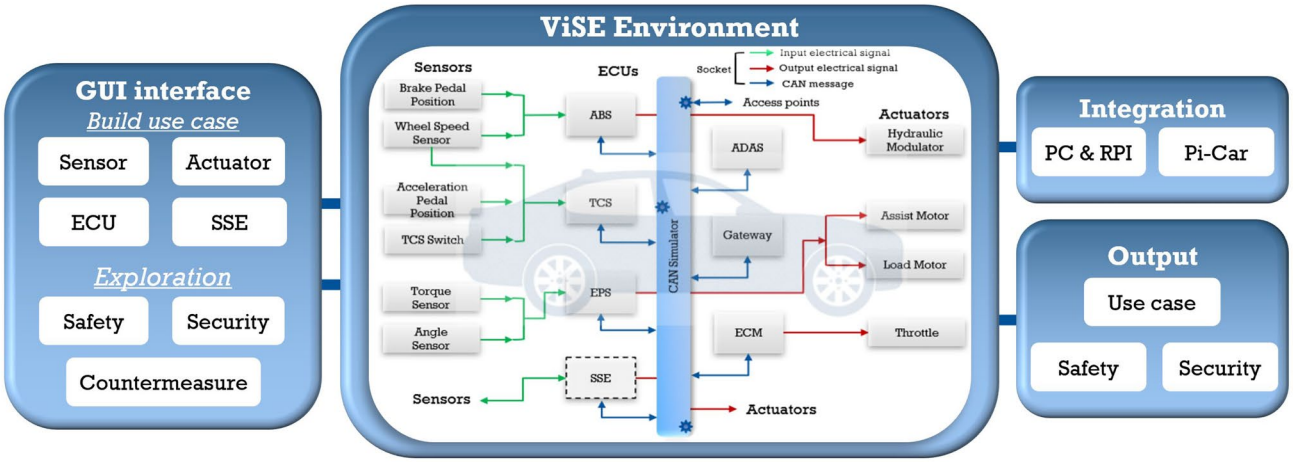

Fig. 1 ViSE digital twin platform architecture involving. The initial interface (left). Platform environment with three representative use cases (center): antilock braking system, traction control system, and right turn. Corresponding hardware integration options and outputs (right)

to address cyberattack concerns. Through the use of virtual reality (VR) technologies, their approach makes it possible for non-experts to perform hands-on exploration of security attacks while learning the implications of these attacks in a comprehensive environment.

There are some closely depicted research works from the avionics industry. Fraser et al. [21] investigated the vulnerability of unmanned aerial vehicles (UAVs) to modern cyber threats, proposing solutions through digital twin architectures and data-driven methods for real-time intrusion detection. It validates these concepts using machine learning models on UAV flight data, demonstrating the effectiveness of DTs in identifying cyber intrusions and anomalies in UAV systems. Another study [22] introduces spatial DTs and a convolutional neural network (CNN) algorithm to enhance the airspace structure and safety of UAV systems, demonstrating improved safety performance and network longevity through advanced clustering algorithms and optimal node management. The findings suggest that the UAV DTs system can significantly boost safety during flight, offering valuable insights for future UAV applications.

ViSE is inspired by—and a substantial extension of—a previous platform developed by Kabir and Ray [2] and Kabir et al. [23]. That platform enabled the exploration of automotive functionality and hardware/software/sensor interactions for system-level use cases. However, the design, scope, and objectives of ViSE are very different from previous work. In particular, while Kabir et al. primarily focused on exploration and optimization, ViSE, unlike tools like Metropolis, adopts a more comprehensive approach by establishing a reconfigurable digital twin framework that enables the targeting of system-level coordination and communication for exploring safety and security aspects.

3 Digital Twin Architecture

The ViSE digital twin platform provides the means to explore, optimize, and evaluate a range of security objectives, with system-level coordination playing a vital role in multiple use cases. While it is possible to abstract the details of vehicular components (beyond the necessary comprehension for the given use cases), vehicular communications, in particular, are integrated into the platform. In this discussion, we will provide a brief overview (see Fig. 1) of our digital twin component models and the significance of the simulation, which plays a pivotal role in visualizing the capabilities of ViSE to encompass functional safety and security considerations.

3.1 Component Models

In automotive systems, the typical configuration includes sensors, actuators, and ECUs interconnected through an in-vehicle communication network (such as direct electrical signals or CAN). To simulate and evaluate use cases and security scenarios, we virtualize all these components within our simulated environment.

3.1.1 ECU

When it comes to ECUs, ViSE takes a different approach compared to conventional simulation platforms. Instead of requiring a complete software model of the ECU, it utilizes the computational capabilities of an actual ECU. It simulates relevant functionality for the use cases by receiving inputs from appropriate sensors or other simulation blocks.

3.1.2 Sensors and Actuators

The platform enables the seamless integration with either physical sensors or software processes to generate computationally generated data that accurately simulates the behavior of automotive subsystem sensors (e.g., wheel speed sensor, acceleration pedal position sensor, pressure sensor). Similar to sensors, actuators are also represented by software processes and primarily serve as outputs from ECUs for actuation activities (e.g., throttle, hydraulic modulator, assist motor). When necessary, the actuator generates GUI blocks that serve as the final output.[1]

3.1.3 SSE Processes

Apart from the usual structural components discussed above for the platform's initial construction, we employed *SSE* (safety and security exploration) processes that are capable of taking part in the simulation of various critical safety and security scenarios. These processes have distinct characteristics, e.g., having direct connections with the user interface, from where the users can tweak parameters to establish component failures, ASIL evaluation, and adversarial attack simulations. Multiple processes are tied to the platform simulation window incorporating various safety and security-related characteristics for simulation. To date, we have successfully implemented simulations of both partial and full component failures (to address safety), as well as frame falsification, DoS attacks, and frame injection (to address security). Future development includes the incorporation of a broader spectrum of sophisticated adversarial attacks.

Remark 1 (Exploration Note) Our goal is to provide a platform for early exploration of safety and security scenarios to enable users to draw insightful conclusions. The focus is not to assess the severity of the security or safety compromises, but rather to explore scenarios and corner cases that have been developed based on certain assessments. Therefore, we do not consider the joint analysis of safety and security properties, as our focus is on examining individual scenarios separately at the system level. Our platform serves as a collaborative space for engineers and security architects to discuss, refine, and potentially anticipate safety and security challenges.

[1] The goal of our digital twin platform is to offer users a real-time comprehension of the interplay between different components and subsystems through automotive use cases. This is achieved by representing all vehicular components as continuously operational computation blocks. For example, in the ABS use case, the brake pedal position sensor consistently transmits brake input data to the ABS ECU.

3.2 Network Simulator

Communication and coordination are integral parts of any digital twin to perform rapidly within the real-time paradigm for accurate exploration. We provide an in-vehicle communication system that allows two types of communication:

1. Electrical signal (without ECU involvement or CAN)
2. CAN communication (among ECUs)

We have integrated a versatile and adaptable communication simulator into our system, known as the "CAN simulator", which accurately replicates the functionalities of an actual automotive CAN bus. To facilitate the interaction between various ECUs, the simulator employs transmission control protocol (TCP) sockets, utilizing socket programming based on a standard client-server model. A byte array message is generated to represent the CAN frame for seamless transmission and reception of data over the socket. This model is implemented using the Python-CAN package. The CAN frame encompasses essential information such as the arbitration ID, extended ID, data length, and actual data. The simulator determines which ECU should receive data based on the arbitration ID in each CAN frame, with each ID assigned to a port number that corresponds to a specific ECU. It adeptly handles incoming messages via sockets, routing them to the correct ECU. Furthermore, it is capable of handling simultaneous messages from multiple ECUs across various use cases using a queuing mechanism, showcasing its robust multitasking capabilities. The CAN simulator is designed to promptly notify the user in case of any network disruptions. For security exploration, the CAN simulator has various access points for adversarial activities to take place as well.

3.3 User Interface

The platform offers a user interface to initiate targeted simulation. The simulation window offers a wide range of options for selecting sensors, ECUs, and actuators to build any desired use case. It enables multiple graphical user interfaces (GUI) to execute actuation actions of pressing and final simulation outputs for thorough analysis and evaluation. In terms of safety and security, the user will be able to navigate within the interface to "tweak" certain parameters and observe various safety and security scenarios.

3.4 Hardware Integration

For a seamless exploration of functionalities using real software, we offer the option to implement all use cases with Raspberry Pi models corresponding to the respective ECUs or an operable pi-car representing all the components. For

Table 1 Right turn use case byte array messages

Components	Data
ABS	Simulated values from [0] to [70] (CAN)
Angle sensor	[0, 90, 180, 270, 360]
Torque sensor	[0] or [3]
EPS (output-1)	Calculated value between [0] and [17]
EPS (output-2)	Calculated value between [0] and [20]
EPS (for gateway)	Output-1 and output-2 values (CAN)

any use case, individual computational blocks for sensors, actuators, and the CAN bus are implemented separately in one device, while the ECUs are realized through RPIs. Each device is assigned an IP address to facilitate integration, resulting in the development of an embedded system with enhanced computational capabilities. Leveraging the ability of RPIs to integrate with actual physical hardware, VISE provides an interface for seamless integration with physical sensors, rather than solely relying on computational processes. Certain security countermeasures can also take place based on hardware integration (e.g., dedicated hardware).

4 Use Cases

To demonstrate the safety and security scenarios effectively, it is crucial to construct representative use cases that allow us to thoroughly examine functional safety and cybersecurity instances. These use cases serve as valuable tools for addressing potential risks and vulnerabilities in our systems. By carefully designing and analyzing these scenarios, we can enhance the overall safety and security of our solutions, ensuring robust protection against potential threats and hazards. The representative use cases discussed here are "right turn" and "cruise control".

4.1 Right Turn

The electric power steering (EPS) system plays a vital role in enabling smooth steering control for vehicles. Specifically, during a right turn, when the driver rotates the steering wheel to the right, the EPS system provides assistance to ensure effortless maneuvering. This functionality is widely standardized [24]. To simulate this use case, our platform incorporates various essential components, including the EPS ECU, ABS ECU, gateway ECU, angle sensor, torque sensor, assist motor, and load motor. Each of these components acts as an indigenous process and communicates via byte array messages using sockets (as depicted in Table 1). The communication among the components is facilitated by unique arbitration IDs (e.g., 1 and 4) and port numbers (e.g., 5004, 7400). The core functionality of this use case, as well as the corresponding system on our platform, is illustrated in Fig. 2a and b, respectively. On the action, the system operates as follows:

- To simulate the driver initiating a right turn, the EPS ECU continuously receives data from the torque sensor and angle sensor. For the purpose of simulation, we assume a steering torque of 3 Nm and consider various steering wheel angles, including 0, 90, 180, 270, and 360°.
- In addition, the EPS ECU receives wheel speed data as a CAN message from the ABS ECU (via arbitration ID 4). This information is necessary for the calculation of assist torque.
- Based on the combined sensory data, the EPS ECU computes the required assist torque. For example, if the load torque required is 20 Nm (which varies with different speed values), the assist torque would be $(20 - 3) = 17$ Nm.
- Subsequently, the EPS ECU transmits the assist torque value to the assist motor and the load torque value to the load motor as commands for their respective actuators.

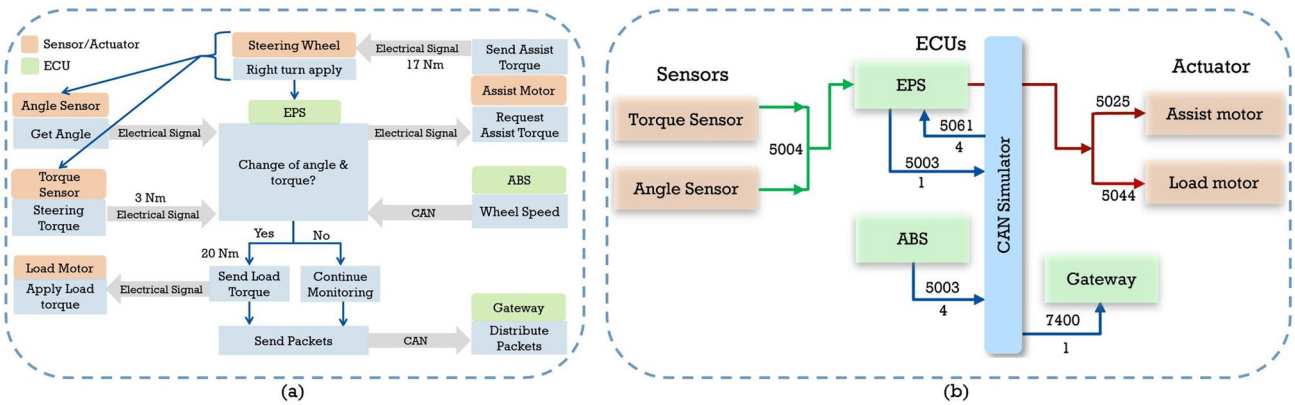

Fig. 2 Right turn (from EPS) use case **a** functionality flow diagram and **b** primary system design

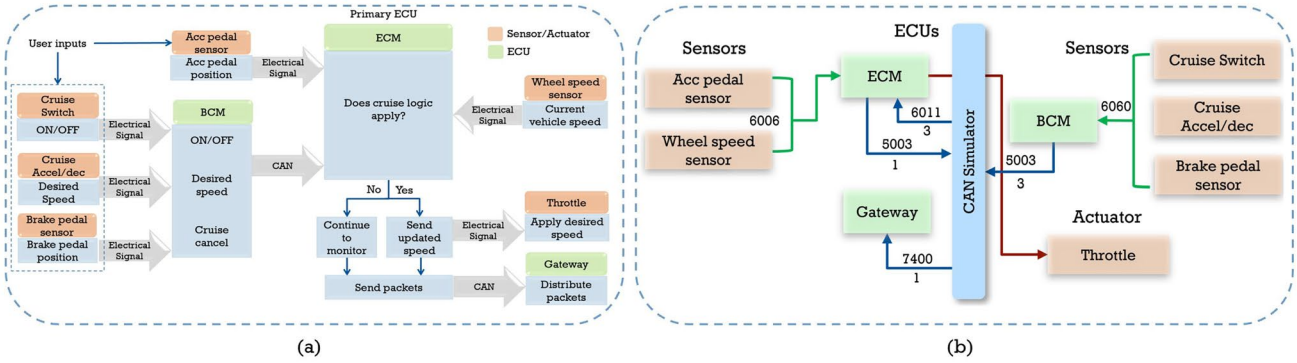

Fig. 3 Cruise control use case **a** functionality flow diagram and **b** primary system design

- Furthermore, the EPS ECU sends a CAN frame back to the CAN simulator, with an arbitration ID of 1, indicating the gateway ECU. The gateway ECU is responsible for forwarding packets to other ECUs and the instrument cluster.

Note that the functionality of the use cases has currently been implemented up to the gateway in order to focus on the system-level activities and driver assistance operations; the instrument cluster (e.g., notifications in the heads-up display) is not implemented.

4.2 Cruise Control

The cruise control system, a crucial feature in modern vehicles, consists of multiple ECUs and parallel activities. To ensure a user-friendly experience, the cruise control functionality is designed to be as straightforward as possible, with minimal interactions required for simulation [25, 26]. In order to simulate this use case, our platform interface provides the representations of the engine control module (ECM), body control module (BCM), gateway ECUs, acceleration pedal position sensor, brake pedal position sensor, wheel speed sensor, cruise ON/OFF switch, cruise acceleration/deceleration buttons, and throttle. Once the cruise control is activated, the user can increase, decrease, or maintain the vehicle speed using the appropriate controls on the GUI. The communication among the components is facilitated by unique arbitration IDs (e.g., 1 and 3) and port numbers (e.g., 6011, 7400). The core functionality of this use case, as well as the corresponding system on our platform, is illustrated in Fig. 3a and b, respectively. On the action, the system operates as follows:

- The cruise function gets activated, i.e., the use case is initiated by the user from the cruise ON/OFF switch.
- The BCM ECU receives the sensory inputs from the brake pedal position sensor, cruise ON/OFF switch, and cruise acceleration/deceleration buttons.
- The ECM ECU seamlessly processes inputs from the acceleration pedal position sensor and receives a comprehensive array of sensory data transmitted by the BCM ECU as a CAN message (via arbitration ID 3).
- As the primary use case ECU, the ECM then processes these inputs, to compute the desired speed and direction for the vehicle and sends the final output to throttle for actuation.
- Subsequently, the ECM sends the current status of the cruise control system as a CAN message to the gateway (via arbitration ID 1). Finally, the gateway assumes the responsibility of routing the packets to other ECUs and the instrument cluster. The dashboard visuals are depicted as GUI output from the gateway itself.
- Since the cruise control system operates continuously, a feedback mechanism is implemented to provide information back to the control system with real-time updates on the cruise control status and the current speed of the vehicle.[2] This entire sequence of receiving inputs, computing the speed, and sending the cruise status occurs within one cycle.

5 Functional Safety Exploration

5.1 Safety Exploration Features

The introduction of functional safety standards aimed to address the risks associated with failures in automotive systems. These standards encompass various aspects, e.g., electronic component failure rates, failure classifications, hardware failure modes, ASIL determination, and safety

[2] With the feedback loop established, subsequent cycles run based on the initial feedback values. This enables the system to continuously update the user interface with the most recent information regarding the cruise control status and the current speed of the vehicle. We are portraying the first cycle only in our discussion here.

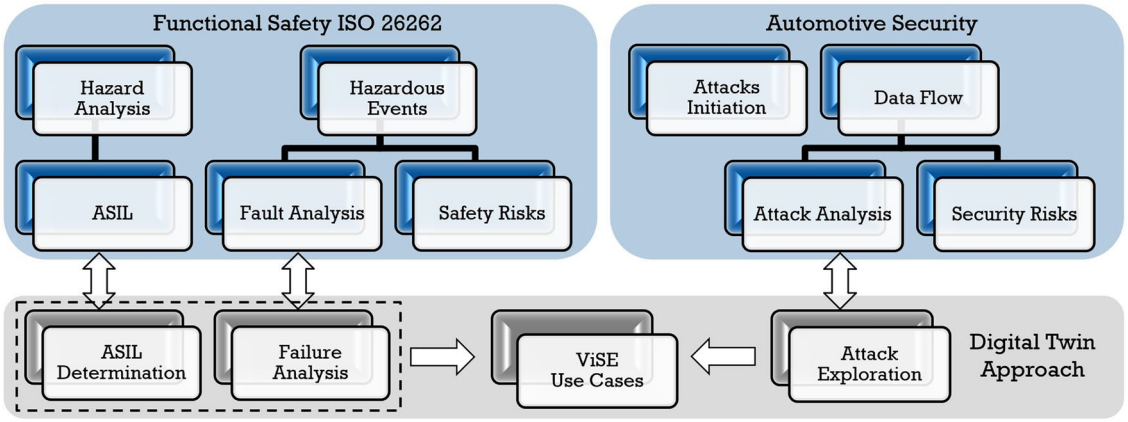

Fig. 4 Proposed digital twin approach for automotive safety and security

requirements. which help comprehend the impact and severity of these failures. Despite these measures, there is a lack of exploration platforms specifically designed for investigating system-level failures within a digital twin environment. To bridge this gap, ViSE offers the following safety features:

- The platform deploys SSE processes (discussed in Section 3.1.3) for safety-related scenario exploration from the user interface. It offers the capability to seamlessly swap active components, deploy blank or partially operating processes, and explore different use cases.
- By integrating hardware such as real sensors and utilizing small-scale computers like Raspberry Pi, we can also enable real-time failure explorations within the virtual environment with less sophistication.
- We address the challenge of ASIL determination by leveraging our innovative digital twin approach that offers the potential to revolutionize ASIL by bridging the gap between assumptions and practicality. As the existing automotive components already have ASIL levels assigned, we show how to explore those and check ASIL compatibility, so that the same method can be used for new systems.
- Several instances of the safety scenario deploy necessary and additional GUIs for the user to comprehend that scenario in detail. The GUIs allow the user to further modify parameters for customized safety scenario explorations.

5.2 Case Studies

We introduce an approach (summarized in Fig. 4) where the digital twin technology can contribute to explore *component failures* and their impact on overall system safety, enabling them to make informed decisions and further enhance automotive functional safety. Additionally, we also show the platform can be useful in the more practical *determination of ASIL levels*. We discuss our approach with ViSE by taking the *right turn* use case as an illustrative example for these case studies.

5.2.1 Component Failure

Automotive component failures can have serious consequences, leading to potential accidents, breakdowns, and even fatalities. Several key components in a vehicle play crucial roles in ensuring its safe and efficient operation. We have introduced the ability to dynamically modify any components, such as ECUs, sensors, or actuators, in a configurable manner, enabling us to examine the performance of corresponding use cases under critical component failures with various failure modes. Figure 5a illustrates a scenario where the load motor actuator either completely fails or loses electrical connection with the electronic power steering (EPS) ECU. Consequently, we simulate the use case by connecting port number 5044 to a blank process, resulting in an error indicating that the wheels are not turning with adequate movement due to the absence of applied torque.

In a real-life situation, this failure scenario would hinder the driver from successfully making a right turn, as there would be either zero or minimal wheel movement (assuming a minimal amount of steering torque provided by the driver). Similarly, a malfunctioning ECU would lead to a severe safety-critical issue, causing significant dysfunction in the ECU's computations.

5.2.2 ASIL Determination

ASIL plays a critical role in the early stages of the development process. It is determined by the hazard analysis and risk assessment (HARA) [27] and relies on a combination of factors, including the probability of exposure (E), controllability (C) by a driver, and severity (S) of potential

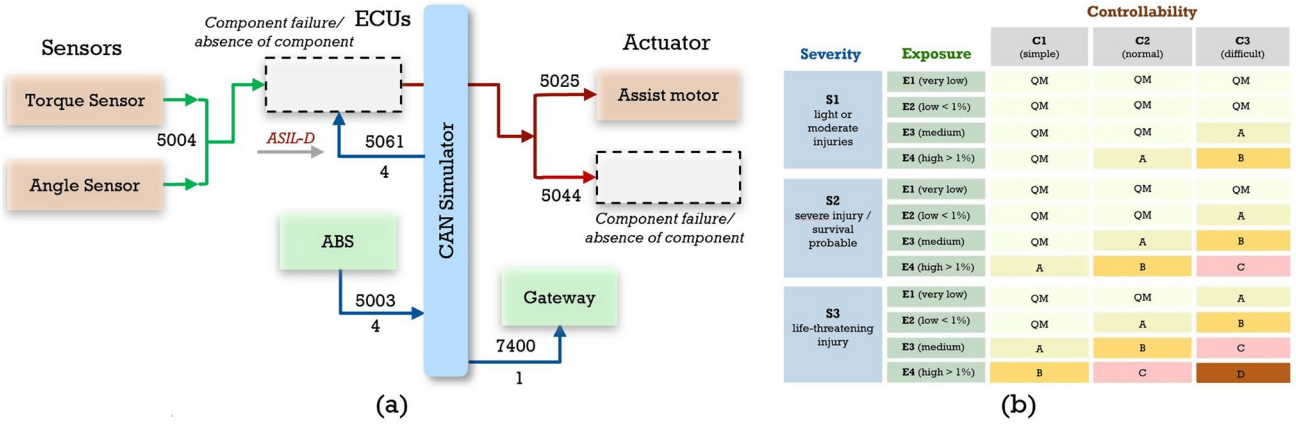

Fig. 5 Safety exploration in right turn use case with ViSE

injuries [28]. The ASIL classification encompasses four levels: ASIL A, B, C, and D, which differ based on subjective assumptions, containing conjectures, human emotions, and cognitive interpretations. In Fig. 5a, the load motor and the EPS ECU are identified as failed components. Since EPS is assigned an ASIL level "D", we will evaluate this determination using ViSE. We understand the following:

- The failure of the EPS ECU could lead to incorrect wheel speed sensor (from ABS), torque sensor, and angle sensor values.
- The *right turn* computations will not take place. Subsequently, the assist motor and load motor will not receive updated data resulting in critical use case dysfunction.
- From Fig. 5b, we understand that this is a life-threatening situation with *severity* level S3. The EPS ECU failure represents the most difficult aspect of *controllability* level C3—as it constitutes a critical failure of the primary ECU.
- The only component our platform will not help identify is *exposure*. Providing this info (E4) from an external source, we ultimately get ASIL D.

Therefore, a user can use the platform to understand the practical implications of different scenarios that can assist in determining the ASIL level much more objectively. However, it is important to note that to achieve accurate results, we require the information of *exposure (E)* as our platform does not assist in determining it.

6 Security Compromise Exploration

6.1 Security Vulnerabilities

The integration of autonomous features in modern automobiles holds tremendous potential for enhancing safety by minimizing human errors. However, this advancement comes with an inherent vulnerability to cyberattacks. Unfortunately, public awareness regarding the security of vehicular systems remains remarkably low despite its critical importance. In adversarial attacks, the initial phase primarily focuses on infiltrating the in-vehicle network, typically achieved by exploiting the onboard diagnostics (OBD) or wireless interfaces [29]. However, alternative entry points have also been explored in various studies, such as leveraging a modified WMA audio file [30] or a USB-connected smartphone [31]. Regardless of the chosen entry point, the primary objective remains the same: gaining access to the CAN bus and subsequently infiltrating the CAN frame, ultimately leading to the disruption of critical functions performed by automotive ECUs. These vulnerabilities underscore the urgent need for robust security measures in autonomous vehicles to effectively mitigate potential cyber threats.

6.2 Security Exploration Features

We discussed various existing security exploration platforms and simulators in Section 2. However, the concept of security from a digital twin perspective, especially at a system level, is relatively new, which prompted us to adopt and explore this approach in the first place. In order to explore adversarial attacks, ViSE offers the following security features:

- The platform deploys SSE processes (discussed in Section 3.1.3) for security-related scenario exploration from the user interface. It helps to simulate the behavior and effects of several adversarial attacks with different use cases.
- The ViSE CAN simulator has various access points and functionalities that enable users to visualize and effectively

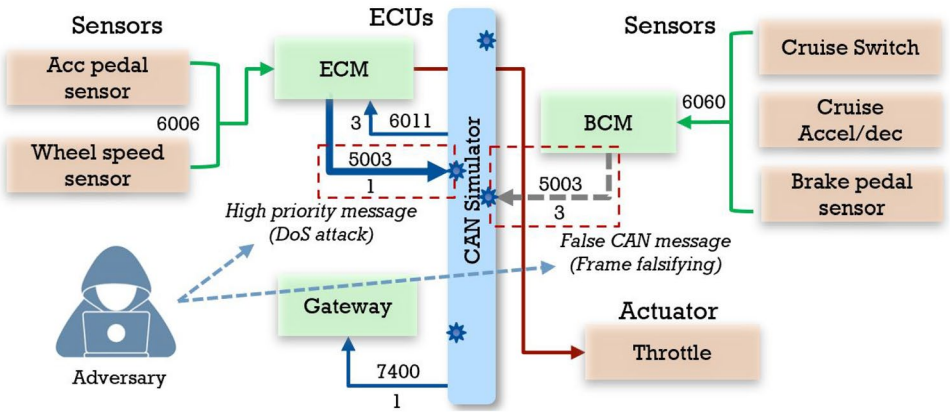

Fig. 6 Security exploration in cruise control use case with ViSE

respond to adversarial attacks. It provides the option to modify CAN messages through the user interface, allowing for simulations, e.g., *frame falsification* to occur.

- To enhance the capabilities of the CAN simulator, we have integrated arbitration IDs and priority schemes, enabling queuing mechanisms to simulate denial-of-service (DoS) attacks [32].
- Several instances of the security scenario deploy necessary and additional GUIs for the user to comprehend that scenario in detail. The GUIs allow the user to further modify parameters for additional security compromise explorations.

6.3 Case Studies

We aim to demonstrate the practicality of our digital twin approach (summarized in Fig. 4) for security exploration through case studies on frame falsification and DoS attacks. We discuss our approach with ViSE by taking the cruise control use case as an illustrative example.

6.3.1 Frame Falsification

Attackers can now plan their attack and send fake frames over the CAN bus since they are aware of the majority of valid CAN frames [33]. The false information in these fake frames has the potential to fool corresponding real ECUs. For instance, the adversary could falsify data from the BCM ECU, affecting the cruise control use case. In Section 4.2, we emphasized the significance of sensory data for accurate cruise control computation in ECM. Let us assume the adversary successfully alters the CAN messages sent from the BCM to the ECM ECU, providing false information about cruise ON/OFF status, acceleration/deceleration, and brake pedal position. Consequently, the ECM's calculations will yield incorrect values for cruise control initiation or disruption. In real-life situations, this can lead to multiple safety-critical problems for the driver: (1) the cruise control may not engage at all, resulting in a false sense of car control, (2) the car may run at an incorrect speed, leading to potential loss of control and serious accidents, or (3) the brakes may not function, causing critical accidents. Our platform offers real-time simulation (Fig. 6) to explore and address such scenarios.

6.3.2 DoS Attack

The automotive system usually employs queuing mechanisms to prioritize ECU messages for transmission. However, this introduces a potential vulnerability, as high-priority frames can lead to the delay or denial of lower-priority messages, making the system susceptible to DoS attacks. Let us consider a scenario (see Fig. 6) where an adversary launches a DoS attack by flooding the CAN simulator with high-priority messages originating solely from the ECM ECU (via port 5003). Consequently, the CAN simulator fails to receive any updates from the BCM ECU, resulting in a lack of crucial sensory data. This situation parallels the safety-critical scenarios discussed earlier in the context of frame falsification, where incorrect data could lead to hazardous consequences. By addressing and exploring such vulnerabilities through our platform, we can better safeguard the system against potential risks and enhance overall security.

Remark 2 (Communication Note) Within this framework, our focus has primarily been on CAN communication, represented through a simplified CAN frame that includes only the essential features from a CAN library to facilitate easier simulation. Other forms of communication have not yet been considered. Therefore, scenarios involving penetration tests, intrusion prevention systems (IPS), local interconnect network (LIN) bus security, etc. are areas we have not explored but plan to address in our future work.

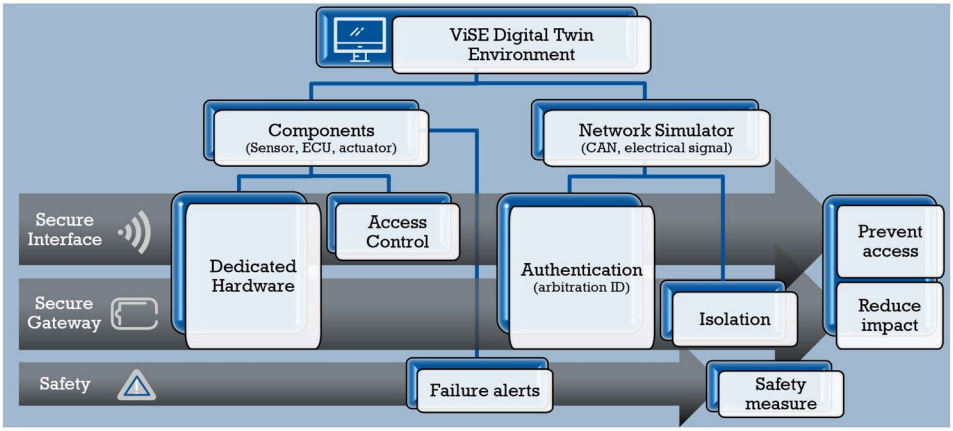

Fig. 7 Countermeasures with ViSE

7 Countermeasures

The automotive industry actively tackles security challenges by incorporating state-of-the-art security principles into automotive design. Protection levels must align with the specific threats within different functional domains, applications, and vehicle components. For instance, the EPS ECU's protection level depends on various factors, e.g., the components involved in right turns, the attack surface, critical functions, and the asset being protected. Components with external connectivity—i.e., the infotainment system or the gateway—require higher levels of protection compared to, e.g., the BCM ECU. To address this issue, the Society of Automotive Engineers (SAE) introduced the cybersecurity guidebook for electronic vehicle systems (SAE J3061) [34]. Additionally, the 2018 edition of ISO 26262 recognizes cybersecurity as a vital aspect of functional safety, highlighting the need to establish a correlation between functional safety and cybersecurity [35].

To showcase the effectiveness of our virtual environment, we explored several safety and security countermeasures, demonstrating its capability to apply core security principles [36]. It is important to note that ViSE primarily serves as an infrastructure for basic failure analysis, rather than a resiliency solution. It does not autonomously implement countermeasures; instead, it supports users in developing and applying their own solutions, empowering them to enhance the security and safety of automotive systems proactively. Our countermeasure approach for safety and security (see Fig. 7) is initiated and divided between the computational blocks (i.e., use case components) and the platform network simulator (i.e., CAN and electrical signal). Some major security countermeasures [19] that prevent access and reduce impact can be exercised in the platform as follows:

- **Dedicated hardware:** To address the scarcity of computing power in ECUs and meet real-time constraints, integrating hardware platforms specifically designed for security functions becomes imperative. In this regard, our platform offers hardware integration options (discussed in Section 3.4) to cater to these requirements effectively.
- **Access control:** To implement secure access control, we employ TCP port numbers as part of our framework. These ports facilitate reliable and controlled communication between ECUs, ensuring only authorized and necessary interactions take place, thus enhancing the overall security and robustness of the system.
- **Authentication:** We employ the usage of arbitration IDs to implement authentication since various ECUs communicate with the CAN simulator at the same time. This countermeasure is inspired by the work of Mundhenk et al. [37].
- **Isolation:** To thwart potential attacks that could damage the entire network, we implement measures to isolate subsystems of one or multiple use cases through the gateway ECU and CAN simulator.

To further enhance our safety and security measures, we implemented several additional steps, detailed below:

- **Safety notification:** Our platform architecture includes a feature that enables the CAN simulator and ECU processes to promptly alert users in the event of specific component failures. This proactive notification system ensures that potential issues are detected at their earliest stage, allowing for timely intervention and preventing more significant problems from arising.
- **Congestion detection scheme:** Congestion in a CAN bus refers to the state where the bus is heavily loaded with messages, leading to potential delays in message transmission. This can happen due to a high volume of legitimate traffic or due to malicious activities. For instance, the frame falsification attack involves maliciously altering or injecting false messages into the CAN bus to disrupt the normal communication flow. We meticulously

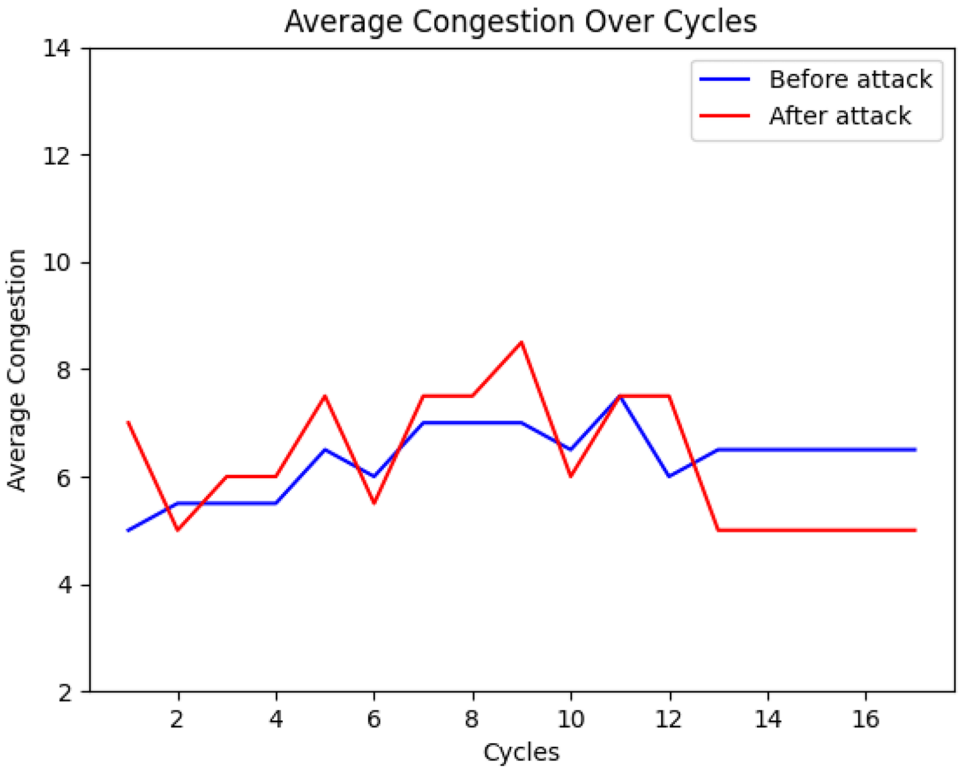

Fig. 8 Congestion changes with frame falsification in ViSE

monitor the congestion levels across various use cases in operation and rigorously track any shifts in congestion patterns following an attack (see Fig. 8).

8 Conclusion

As automotive systems become increasingly complex, exploring and validating system-level functional safety and security scenarios during the early stages of automotive system design is becoming of vital importance. In this paper, we developed a platform, ViSE, that allows exploration based on real-life attack occurrences and failure modes. We illustrated the effectiveness of the platform with two representative system-level use cases: *right turn* and *cruise control*. Additionally, we established the foundation for security policies and countermeasures that can be freely explored within the virtual environment.

In future work, we will expand ViSE to provide an executable prototyping model that is fully compliant with all aspects of the ISO 26262 standard. We also plan to develop new security use cases involving the interaction of hardware, sensor, and software modules that target autonomous functionalities in emergent vehicles. In particular, many emergent cyberattacks in autonomous vehicles involve the interaction of computer vision modules, artificial intelligence, and V2X communications, which will be interesting to comprehend and explore in ViSE.

Statements and Declarations

Funding This research has been supported in part by the National Science Foundation under Grant No. CNS-1908549 and SATC-2221900.

Conflict of Interest The authors declare no competing interests.

Author Contribution Both authors contributed to the research, and they reviewed and edited the manuscript.

Data Availability Not applicable.

Ethical Approval Not applicable.

References

1. Da Xu L, He W, Li S (2014) Internet of Things in industries: a survey. IEEE Trans Industr Inf 10(4):2233–2243
2. Kabir MR, Ray S (2023) Virtualization for automotive safety and security exploration. 2023 IEEE 16th Dallas Circuits and Systems Conference (DCAS), Denton, TX, USA, pp 1–4. https://doi.org/10.1109/DCAS57389.2023.10130221
3. Wagg DJ, Worden K, Barthorpe RJ, Gardner P (2020) Digital twins: state-of-the-art and future directions for modeling and simulation in engineering dynamics applications. ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part B: Mechanical Engineering 6(3):030901
4. Grieves MW (2019) Virtually intelligent product systems: digital and physical twins. pp 175–200
5. Barricelli BR, Casiraghi E, Fogli D (2019) A survey on digital twin: definitions, characteristics, applications, and design

implications. IEEE Access 7:167653–167671. https://doi.org/10.1109/ACCESS.2019.2953499
6. Boschert S, Rosen R (2016) Digital twin–the simulation aspect. Mechatronic Futures. Springer, pp 59–74
7. Qi Q, Tao F, Hu T, Anwer N, Liu A, Wei Y et al (2021) Enabling technologies and tools for digital twin. J Manuf Syst 58:3–21
8. Kabir MR, Ray S (2023) Virtual prototyping for modern Internet-of-Things applications: a survey. IEEE Access 11:31384–31398. https://doi.org/10.1109/ACCESS.2023.3262499
9. Almeaibed S, Al-Rubaye S, Tsourdos A, Avdelidis NP (2021) Digital twin analysis to promote safety and security in autonomous vehicles. IEEE Commun Stand Mag 5(1):40–46. https://doi.org/10.1109/MCOMSTD.011.2100004
10. Association IS et al (2020) IEEE draft standard for transparency of autonomous systems. IEEE P7001(D1):1–70
11. Damjanovic-Behrendt V (2018) A digital twin-based privacy enhancement mechanism for the automotive industry. 2018 International Conference on Intelligent Systems (IS). IEEE, pp 272–279
12. Shadrin S, Makarova D, Ivanov A, Maklakov N (2021) Safety assessment of highly automated vehicles using digital twin technology. 2021 Intelligent Technologies and Electronic Devices in Vehicle and Road Transport Complex (TIRVED). IEEE, pp 1–5
13. Safar M, El-Moursy MA, Abdelsalam M, Bakr A, Khalil K, Salem A (2019) Virtual verification and validation of automotive system. J Circuit Syst Comp 28(04):1950071
14. Behrisch M, Bieker L, Erdmann J (2011) SUMO–simulation of urban mobility: an overview. Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation. ThinkMind
15. Dosovitskiy A, Ros G, Codevilla F, Lopez A, Koltun V (2017) CARLA: an open urban driving simulator. Conference on Robot Learning. PMLR, pp 1–16
16. Zeng H, Davare A, Sangiovanni-Vincentelli A, Sonalkar S, Kanajan S, Pinello C (2006) Design space exploration of automotive platforms in metropolis. SAE Transactions, pp 844–856
17. Yang C, Dong J, Xu Q, Cai M, Qin H, Wang J (2022) Multi-vehicle experiment platform: a digital twin realization method. 2022 IEEE/SICE International Symposium on System Integration (SII). IEEE, pp 705–711
18. Ravi BBY, Kabir MR, Mishra N, Boddupalli S, Ray S (2022) Autohal: an exploration platform for ranging sensor attacks on automotive systems. 2022 IEEE International Conference on Consumer Electronics (ICCE). IEEE, pp 1–2
19. Scalas M, Giacinto G (2019) Automotive cybersecurity: foundations for next-generation vehicles. 2019 2nd International Conference on new Trends in Computing Sciences (ICTCS). IEEE, pp 1–6
20. Owoputi R, Kabir MR, Ray S (2023) IVE: An immersive virtual environment for automotive security exploration. Immersive Learning Research - Academic 1(1):468–480
21. Fraser B, Al-Rubaye S, Aslam S, Tsourdos A (2021) Enhancing the security of unmanned aerial systems using digital-twin technology and intrusion detection. 2021 IEEE/AIAA 40th Digital Avionics Systems Conference (DASC). IEEE, pp 1–10
22. Wang W, Li X, Xie L, Lv H, Lv Z (2021) Unmanned aircraft system airspace structure and safety measures based on spatial digital twins. IEEE Trans Intell Transp Syst 23(3):2809–2818
23. Kabir MR, Ravi BBY, Ray S (2023) A virtual prototyping platform for exploration of vehicular electronics. IEEE Internet Things J. https://doi.org/10.1109/JIOT.2023.3267339
24. Kim JH, Song JB (2002) Control logic for an electric power steering system using assist motor. Mechatronics 12(3):447–459
25. Shaw M (1995) Beyond objects: a software design paradigm based on process control. ACM SIGSOFT Software Engineering Notes 20(1):27–38
26. Girovský P, Fekete J (2017) Maintaining vehicle speed using a mechanical cruise control. Acta Electrotechnica et Informatica 17(2):48–52
27. ISO 26262 "Road Vehicles –Functional Safety" (2018) International Organization for Standardization. https://www.iso.org/standard/68383.html
28. Jeon SH, Cho JH, Jung Y, Park S, Han TM (2011) Automotive hardware development according to ISO 26262. 13th International Conference on Advanced Communication Technology (ICACT2011). IEEE, pp 588–592
29. Liu J, Zhang S, Sun W, Shi Y (2017) In-vehicle network attacks and countermeasures: challenges and future directions. IEEE Network 31(5):50–58
30. Checkoway S, McCoy D, Kantor B, Anderson D, Shacham H, Savage S, Koscher K, Czeskis A, Roesner F, Kohno T (2011) Comprehensive experimental analyses of automotive attack surfaces. In 20th USENIX Security Symposium (USENIX Security 11)
31. Mazloom S, Rezaeirad M, Hunter A, McCoy D (2016) A security analysis of an in-vehicle infotainment and app platform. In 10th USENIX Workshop on Offensive Technologies (WOOT 16)
32. Palanca A, Evenchick E, Maggi F, Zanero S (2017) A stealth, selective, link-layer denial-of-service attack against automotive networks. Detection of Intrusions and Malware, and Vulnerability Assessment: 14th International Conference, DIMVA 2017, Bonn, Germany, July 6–7, 2017, Proceedings 14. Springer, pp 185–206
33. Philipsen SG, Andersen B, Singh B (2021) Threats and attacks to modern vehicles. IEEE International Conference on Internet of Things and Intelligence Systems (IoTaIS). IEEE, pp 22–27
34. SAE International (2016) Cybersecurity guidebook for cyber-physical vehicle systems. SAE Standard J3061. Available from: https://www.sae.org/standards/content/j3061_201601/
35. Xie G, Li Y, Han Y, Xie Y, Zeng G, Li R (2020) Recent advances and future trends for automotive functional safety design methodologies. IEEE Trans Industr Inf 16(9):5629–5642. https://doi.org/10.1109/TII.2020.2978889
36. Simacsek B (2019) Can we trust our cars. NXP Semiconductors-Paper. Available from: https://www.nxp.com/docs/en/white-paper/AUTOSECWP.pdf
37. Mundhenk P, Paverd A, Mrowca A, Steinhorst S, Lukasiewycz M, Fahmy SA et al (2017) Security in automotive networks: lightweight authentication and authorization. ACM Transactions on Design Automation of Electronic Systems (TODAES) 22(2):1–27. https://doi.org/10.1145/2960407