Exploiting Diagnostic Protocol Vulnerabilities on Embedded Networks in Commercial Vehicles

Rik Chatterjee
Colorado State University
rik.chatterjee@colostate.edu

Carson Green
Colorado State University
carson.green@colostate.edu

Jeremy Daily
Colorado State University
jeremy.daily@colostate.edu

***Abstract*—Modern automotive operations are governed by embedded computers that communicate over standardized protocols, forming the backbone of vehicular networking. In the domain of commercial vehicles, these systems predominantly rely on the high-level protocols running on top of the Controller Area Network (CAN) protocol for internal communication in medium and heavy-duty applications. Critical to this ecosystem is the Unified Diagnostics Services (UDS) protocol, outlined in ISO 14229 (Unified Diagnostic Services - UDS) and ISO 15765 (Diagnostic Communication over CAN), which provides essential diagnostic functionalities. This paper presents three distinct scenarios, demonstrating potential shortcomings of the UDS protocol standards and how they can be exploited to launch attacks on in-vehicle computers in commercial vehicles while bypassing security mechanisms.**

In the initial two scenarios, we identify and demonstrate two vulnerabilities in the ISO 14229 protocol specifications. Subsequently, in the final scenario, we highlight and demonstrate a vulnerability specific to the ISO 15765 protocol specifications.

For demonstration purposes, bench-level test systems equipped with real Electronic Control Units (ECUs) connected to a CAN bus were utilized. Additional testing was conducted on a comprehensively equipped front cab assembly of a 2018 Freightliner Cascadia truck, configured as an advanced test bench. The test results reveal how attacks targeting specific protocols can compromise individual ECUs. Furthermore, in the Freightliner Cascadia truck setup, we found a network architecture typical of modern vehicles, where a gateway unit segregates internal ECUs from diagnostics. This gateway, while designed to block standard message injection and spoofing attacks, specifically allows all UDS-based diagnostic messages. This selective allowance inadvertently creates a vulnerability to UDS protocol attacks, underscoring a critical area for security enhancements in commercial vehicle networks. These findings are crucial for engineers and programmers responsible for implementing the diagnostic protocols in their communication subsystems, emphasizing the need for enhanced security measures.

I. Introduction

Medium and heavy-duty (MHD) vehicles form an essential pillar of the US's critical infrastructure, playing a vital role in transporting goods and supporting various services, including emergency response. The evolution of MHD vehicles towards higher levels of electronification has led to most mechanical operations being controlled through embedded computers, known as electronic control units (ECUs). These ECUs, interconnected within the vehicle through a bus topology network, handle mission-critical information crucial for the vehicle's functionality and safety. In MHD vehicles, the primary communication specifications within these networks are based on the SAE J1939 standard [1]. The specifications of the SAE J1939 are structured in layers, akin to the ISO/OSI [2] standards prevalent in traditional IT networking. The foundational physical layers of the SAE J1939 standards employ the Controller Area Network (CAN) specifications [3] to facilitate the in-vehicle information exchange, a system widely used across automotive networking.

While the robustness and resilience of CAN in automotive networking are well-established, its security aspects, particularly in MHD vehicles, warrant closer examination. Vulnerabilities at various entry points—both remote and local—have been demonstrated to allow control or disruption of vehicle operations [4], [5], [6]. Moreover, the SAE J1939 protocols themselves, while instrumental in the cyber-physical functioning of these vehicles, have been shown to be susceptible to cyber-attacks [7], [8], [9], [10]. Previous works have focused on various layers of the SAE J1939 standards, from the application layer to the network management and data-link layers, uncovering specific vulnerabilities.

However, a critical aspect that remains less explored is the role of diagnostic protocols, particularly the Unified Diagnostic Services (UDS) standards outlined in ISO 14229 [11] and ISO 15765 [12]. Diagnostics play a crucial role in maintaining vehicle health, safety, and efficiency. UDS provides a standardized approach to diagnostic services, which includes vehicle diagnostics, programming, and fault resolution. Given the integral role of diagnostics in MHD vehicles, vulnerabilities within the UDS standards can have significant implications, potentially affecting vehicle performance, safety, and the integrity of the entire vehicular network.

While previous research efforts have predominantly concentrated on uncovering and exploiting vulnerabilities in authentication mechanisms of diagnostic protocols [6], [13], [14], there exists a critical gap in addressing inherent vulnerabilities within the diagnostic protocols themselves. Our research aims to bridge this gap by delving deeper into the latent vulnerabilities present in these protocols. In this paper, we shift the

Symposium on Vehicles Security and Privacy (VehicleSec) 2024
26 February 2024, San Diego, CA, USA
ISBN 979-8-9894372-7-6
https://dx.doi.org/10.14722/vehiclesec.2024.23046
www.ndss-symposium.org

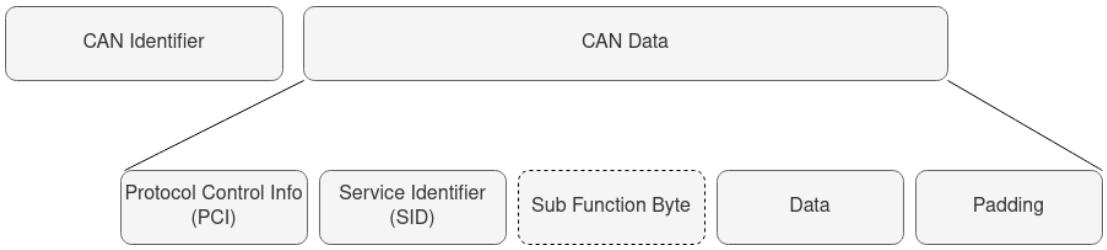


Fig. 1: UDS Message Structure

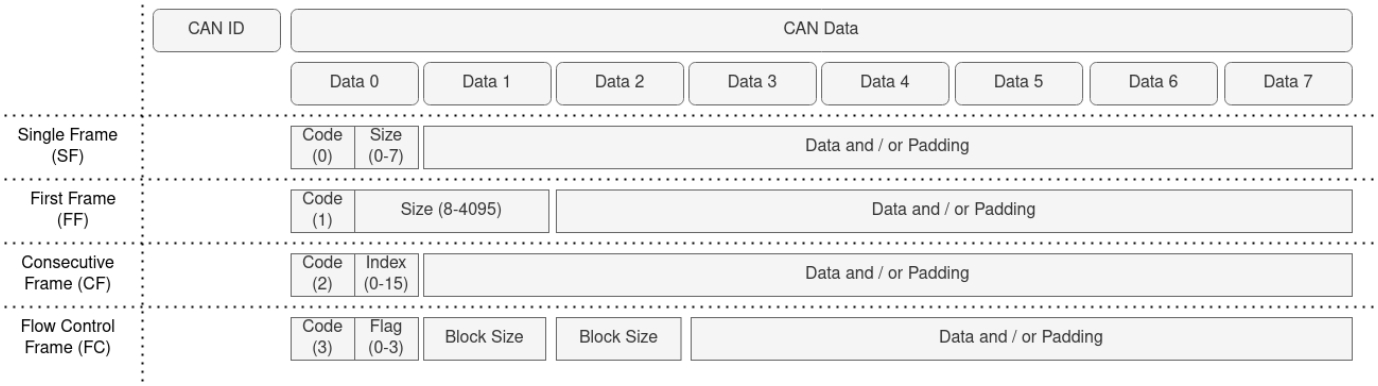


Fig. 2: Different UDS Messages

focus to explore and demonstrate protocol vulnerabilities in the Unified Diagnostic Services (UDS) standards as applied in MHD vehicles. This exploration diverges from previous research that predominantly concentrated on seed-key exchange issues in UDS for passenger cars. We instead delve into the inherent weaknesses within the UDS protocol specifications, aiming to uncover and highlight attack vectors that are not dependent on exploiting the seed-key length vulnerabilities.

Our investigation also reveals a critical aspect of the network architecture in a Freightliner Cascadia cab setup, shown in Fig. 4. This testbed contains most of the wiring and electronics in the cab of the common Freightliner Cascadia heavy truck. Here, the segregation of internal ECUs from diagnostic interfaces through a gateway unit presented a unique scenario: while the gateway effectively shields the internal network from standard message injection and spoofing attacks, it allows all diagnostic messages to pass through. This finding is significant as it suggests that diagnostic-based attacks, which are generally permitted by the gateway, could potentially exploit vulnerabilities in the UDS protocol, thus posing a new threat vector in MHD vehicle networks. Our research, therefore, underscores the necessity of re-evaluating security strategies, particularly focusing on the security of diagnostic communications and the role of gateway configurations in MHD vehicles.

This paper aims to broaden the understanding of the threatscape for in-vehicle networking applications in MHD vehicles, emphasizing the critical need for robust security measures in the face of evolving cyber threats. The remainder of the paper is organized as follows: Section II presents a brief overview of the protocol standards relevant to this research, Section III covers related work in this domain, Section IV describes our testing setup, Section V details the attack experimentation and findings, and Section VI concludes with final remarks and future work considerations.

II. Background

The communication systems in medium and heavy-duty (MHD) vehicles rely on a set of important protocols. These include SAE J1939 over Controller Area Network (CAN), ISO 14229 (Unified Diagnostic Services - UDS), and ISO 15765 (Diagnostic Communication over CAN). Each of these protocols serves a special role, from handling everyday vehicle functions to managing diagnostics and safety. In this section, we will take a closer look at each of these protocols to understand how they enable communication in MHD vehicles.

A. SAE J1939

In-vehicle communication in medium and heavy-duty vehicles is mostly guided by the SAE J1939 standards over the physical Controller Area Network (CAN). SAE J1939 messages carry operational parameters like engine speed, vehicle speed, switch status, etc. These parameters are bundled into logical groups referred to as Parameter Groups (PG). Each PG is identified by a unique number called a Parameter Group Number (PGN), which is also embedded in the message. Information in the J1939 message is carried in a J1939 Protocol Data Unit (PDU). A J1939 PDU also bears a source address (SA) identifying the sender, a destination address (DA) identifying the receiver, the priority of the message, the PGN, and up to 1785 bytes of data. The priority, PGN, SA, and DA

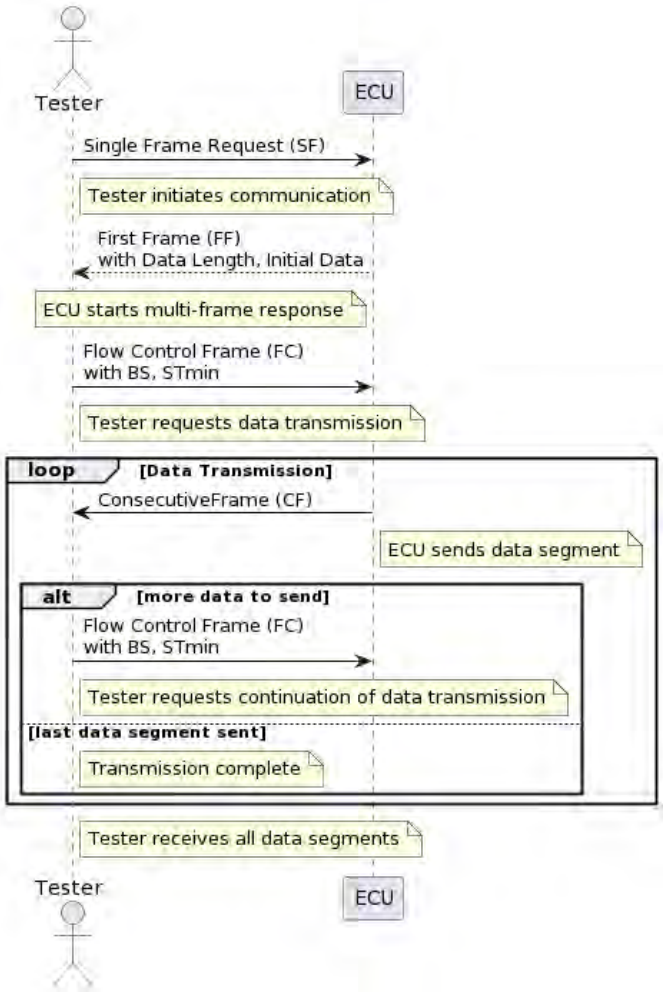


Fig. 3: Logical Point-to-Point Multiframe Data Transfer using ISO-TP

Fig. 4: Freightliner Cascadia Cab Testbed

are embedded into the identifier (ID) part of a CAN frame for PDUs with 8 or fewer bytes. For PDUs that have more than 8 bytes, a transport protocol (TP) is used, and the PGN is located in the last 3 bytes of the TP Connection Management (CM) message data. In case of diagnostic messages, the SAE J1939 specifies a special PGN, 55808 (`0x0DA00`).

TABLE I: Common Service Identifiers in UDS

SID	Service	Positive Response
0x10	Diagnostic Session Control	0x50
0x11	ECU Reset	0x51
0x14	Clear Diagnostic Information	0x54
0x19	Read DTC Information	0x59
0x22	Read Data by Identifier	0x62
0x27	Security Access	0x67
0x28	Communication Control	0x68
0x2E	Write Data by Identifier	0x6E
0x31	Routine Control	0x71
0x3E	Tester Present	0x7E
Common Negative Response		0x7F

B. ISO 14229: Unified Diagnostic Services

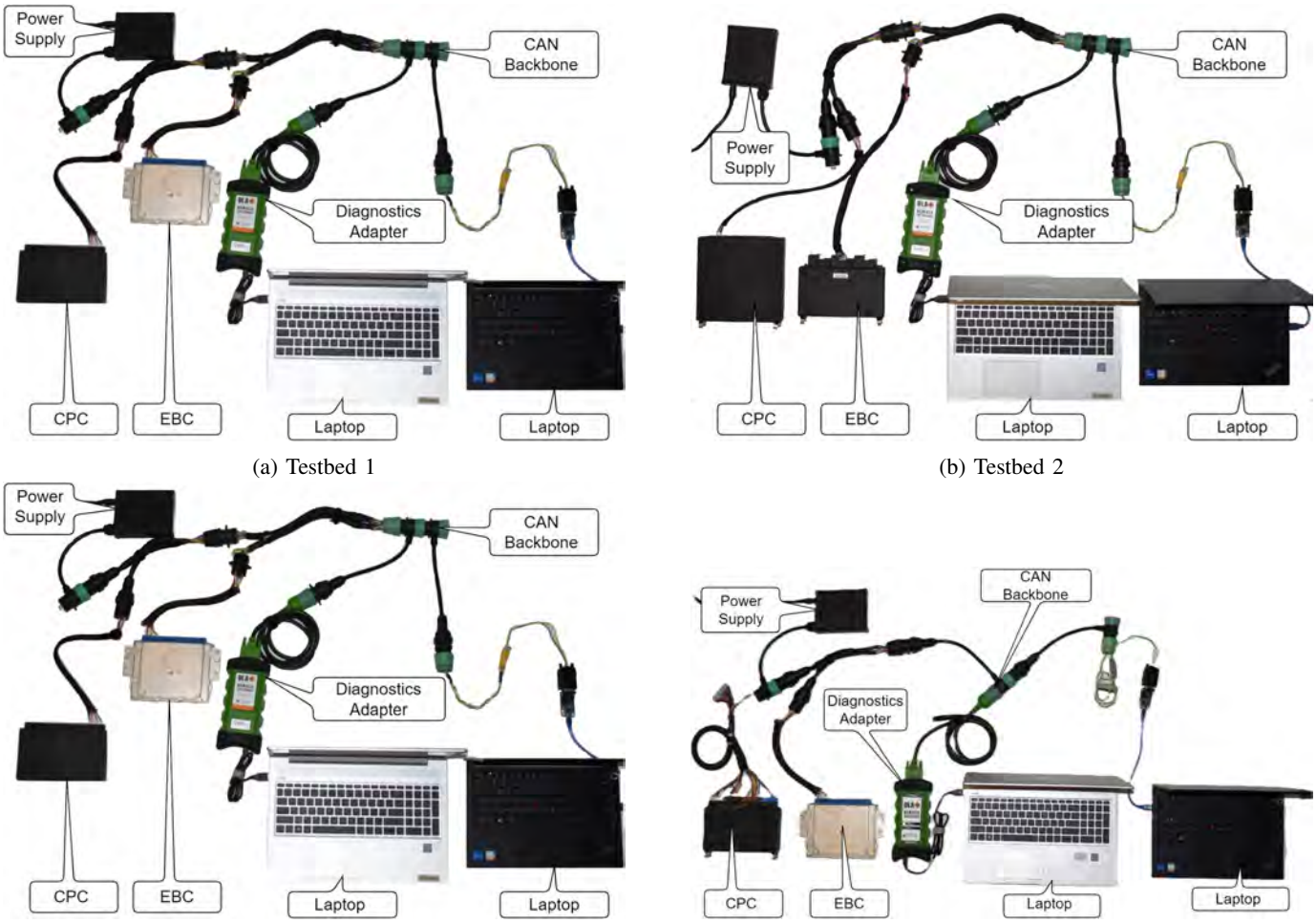
ISO 14229, commonly referred to as Unified Diagnostic Services (UDS), is a critical protocol in automotive diagnostics, facilitating communication between a vehicle's Electronic Control Units (ECUs) and external diagnostic tools. Central to UDS are several key components that structure the diagnostic communication process as shown in Fig. 1. The Protocol Control Info (PCI) serves the role of identifying the type of UDS message, the size of the message, or other identifying parameters of the message. Accompanying the PCI is the Service Identifier (SID), which is pivotal in specifying the type of diagnostic service or function being requested or performed. Additionally, UDS messages often include a Sub-function field, providing further detail or instructions related to the diagnostic service. The Data segment of the message then carries the specific information or command pertinent to the service request. This structured approach enables a wide range of diagnostic operations, including reading or writing data, testing functions, and acquiring information about ECUs or the vehicle.

To illustrate the application of SIDs within UDS, Table I presents common service identifiers, along with typical request codes, positive response codes, and codes for negative responses.

C. ISO 15765: Diagnostic Communication over CAN

ISO 15765 is an automotive standard for communication over the Controller Area Network (CAN), especially for diagnostics. As shown in Fig. 3, it handles the transmission of data packets larger than the limit of a single CAN frame, which is crucial for tasks like detailed diagnostics and ECU programming. Data in ISO 15765 is transmitted in different frame types as shown in Fig. 2, each with a specific format and purpose:

1) Single Frame (SF): Used for data up to 7 bytes. It starts with 0 in the first nibble, followed by the data length in



(a) Testbed 1

(b) Testbed 2

(c) Testbed 3

(d) Testbed 4

Fig. 5: Four Local Testbed Setups

the next nibble (half-byte). The remaining bytes carry the actual data.

2) First Frame (FF): Initiates a multi-frame transmission for data exceeding 7 bytes. It begins with 1 in the first nibble, followed by the length of the total data payload in the next 3 nibbles (12-bits). The rest of the frame contains the beginning portion of the data.
3) Consecutive Frame (CF): Transports subsequent data chunks after the first frame. Each CF starts with 2 in the first nibble, followed by a frame number that increments with each consecutive frame in the second nibble. This number helps in keeping track of the sequence of the frames.
4) Flow Control Frame (FC): Manages the rate of data transmission in a multi-frame message. It begins with 3, and the next nibble indicates the flow status: 0 for Continue To Send (CTS), 1 for Wait, and 2 for Overflow/Abort. The CTS allows the transmission to proceed, Wait tells the sender to pause, and Overflow/Abort indicates that the receiver cannot handle more data. The frame also specifies how many frames can be sent at a time and the time gap between frames.

These frames collectively enable the transmission of large data packets over CAN. Single frames are for small data packets, while the combination of first, consecutive, and flow control frames manages larger data transfers efficiently and reliably.

III. Related Work

The security aspects of automotive protocols, though critical, have historically been underemphasized in research. Recent efforts, however, have begun to uncover multiple vulnerabilities within these protocols.

Kosher et al. [15] shed light on the vulnerabilities in seed-key exchanges among ECUs in passenger cars, revealing that commonly used 8 or 16-bit seed-key pairs are susceptible to brute-force attacks. This finding raises concerns about the ease with which authenticated security sessions can be compromised, potentially granting unauthorized access to critical ECU functionalities. Expanding on this, Miller et al. [6] demonstrated the feasibility of breaking security authentication to ECUs in passenger cars. Through reverse engineering the firmware of diagnostic software in a Ford Escape and a Toyota

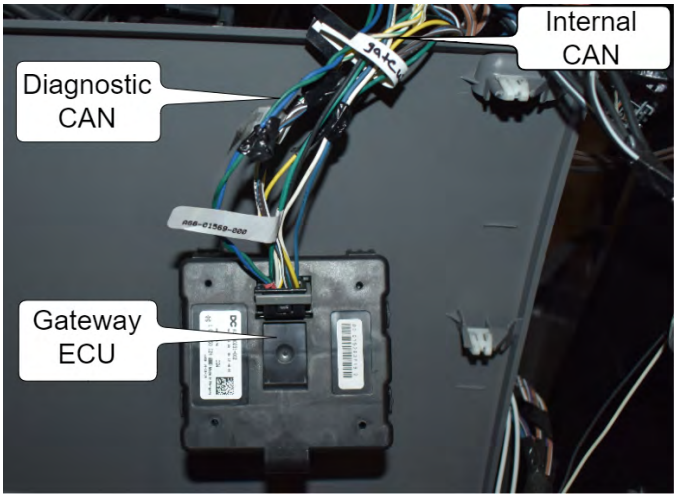


Fig. 6: Gateway Unit in Cascadia Testbed

Prius, they discovered the algorithm used for key calculation from seeds. Their research illustrated several practical attacks, including manipulating brakes, lights, and engine functions, thereby highlighting security gaps.

Burakova et al. [7] identified vulnerabilities within the application layer of the SAE J1939 Protocol. Their work demonstrated the possibility of gaining continuous control over a truck's engine by utilizing specific J1939 messages. Furthermore, they showed how essential functionalities, such as engine braking and accelerator input, could be compromised, underlining the potential dangers in commercial vehicle operations. Mukherjee et al. [8] focused on the data-link layer protocols, revealing how rapid request messages to an ECU could overload its processing capabilities. They also observed that illegitimately sustained connections with an ECU could preclude legitimate connections, presenting a nuanced form of denial of service. In the realm of network management, Murvay et al. [9] highlighted how ECUs could be rendered inoperative by flooding the network with specific address claim messages. They also demonstrated a denial-of-service attack through the abrupt termination of multi-packet data transfers. Mukherjee, et al. verified the address claim vulnerability and proposed a real-time mitigation [16].

Maag et al. [13] contributed to understanding the cybersecurity shortcomings in seed-key exchanges between ECUs and vehicle diagnostics adapters (VDAs) in MHD vehicles. Their work indicated a linear mapping in seed-key pairs, making it feasible to predict these pairs in 16-bit configurations. Lastly, Kulandaivel et al. [14] uncovered multiple vulnerabilities in the UDS implementation in passenger cars. His research was primarily focused on gaining secured access to ECUs, revealing that seeds used in challenge-response pairs are not entirely random and can be influenced by ECU uptime. This insight into predictable seed generation further emphasizes the vulnerabilities in automotive security protocols.

IV. Experimental Testing Setup

To ensure consistent results across our experiments, we conducted tests on multiple configurations of a local bench and a comprehensive testbed assembly of a 2018 Freightliner Cascadia cab. Each testbed contained at least one ECU that communicated using UDS. This section outlines the various configurations of the local testbeds and the Freightliner Cascadia testbed.

A. Bench Testbed Configurations

The local bench testbeds were set up in four distinct configurations as shown in Fig. 5, each involving different combinations of components. Each testbed had a target ECU for testing our vulnerabilities and another control ECU.

1) **Testbed 1:** This setup included a Bendix EC-80 Electronic Brake Controller (EBC) paired with a Detroit Diesel CPC 3 (Common Powertrain Controller), operating on a 250kbps CAN bus. The CPC 3 had a Controller Application (CA) with an address of 0 (`0x00`), while the brakes were assigned the address 11 (`0x0B`). The Bendix EC-80 EBC was the target ECU for this testbed.
2) **Testbed 2:** This configuration incorporated a Wabco Smarttrac system coupled with a CPC 3 EVO, operating on a faster 500kbps CAN bus. The controller addressing scheme remained consistent with the other testbeds, with the CPC 3 EVO having an address of 0 (`0x00`) and the EBC having an address of 11 (`0x0B`). The Wabco Smarttrac EBC was the target ECU for this testbed.
3) **Testbed 3:** This configuration was similar to **Testbed 1** and included a Bendix EC-80 Electronic Brake Controller (EBC) paired with a Detroit Diesel CPC 3 (Common Powertrain Controller), operating on a 250kbps CAN bus. The CPC 3 had a Controller Application (CA) with an address of 0 (`0x00`), while the brakes were assigned the address 11 (`0x0B`). The CPC 3 was the target ECU for this testbed.
4) **Testbed 3:** This configuration also featured a Bendix EC-80 EBC, but with a Detroit Diesel CPC 4, operating on a similar 250kbps bus. The addressing scheme was akin to Testbed 1, with the CPC having a CA with address 0 (`0x00`) and the EBC having an address of 11 (`0x0B`). The CPC 4 was the target ECU for this testbed.

Each testbed was equipped with a Linux laptop running SocketCAN and the 'can-utils' software for capturing and transmitting CAN messages. The laptops also served as points for initiating attacks. Additionally, each testbed included one or more power supplies. For diagnostic communication, a separate laptop equipped with Noregon DLA as the RP1210 compliant vehicle diagnostics adapter (VDA), interfacing with the ECUs in the testbeds.

B. Freightliner Cascadia Testbed

The Freightliner Cascadia testbed, shown in Fig. 4, specifically focusing on the front cab of a 2018 model, was utilized for additional demonstration of our findings. The configuration

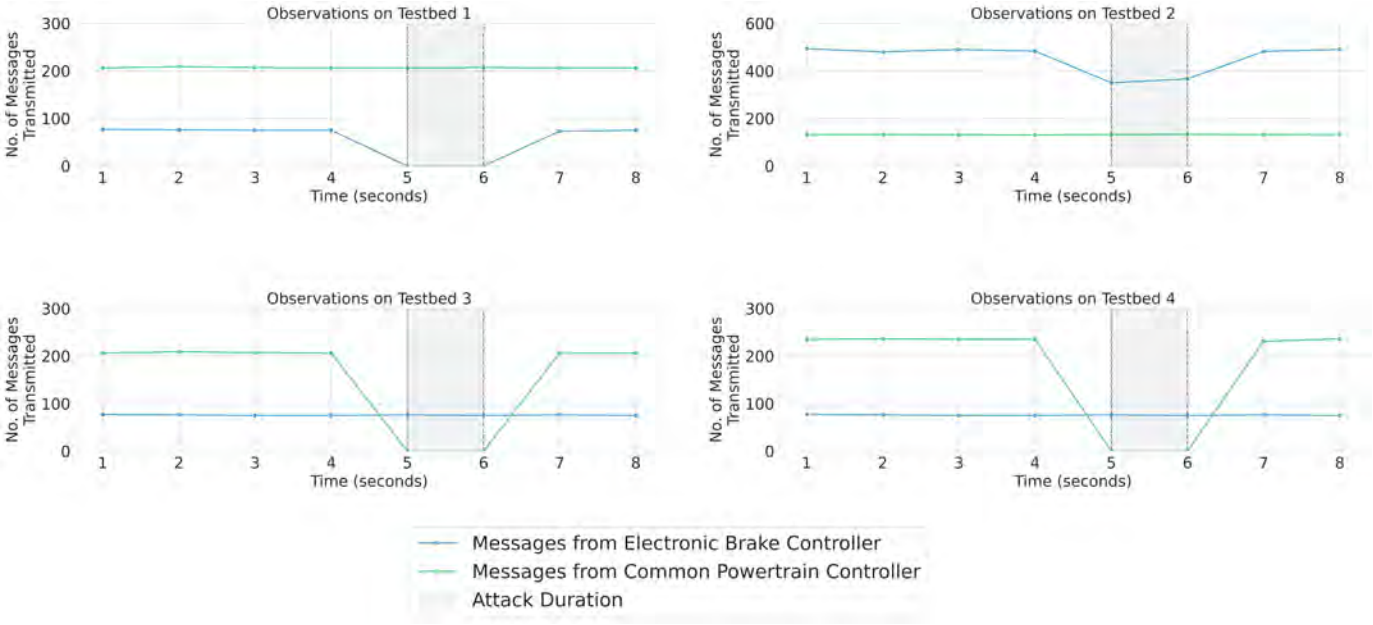


Fig. 7: Read Data by ID Overload Attack at 0.3 ms Interval Attack Messages

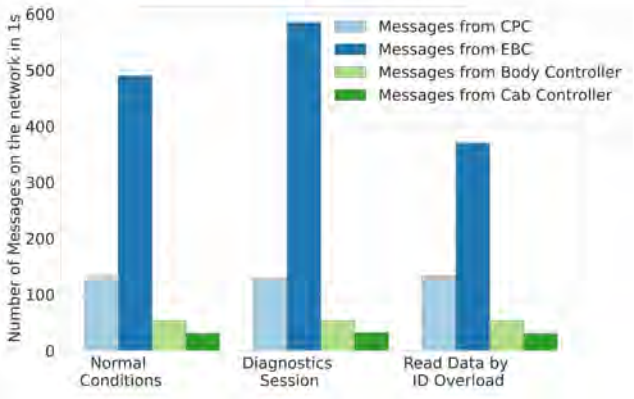


Fig. 8: Read Data by ID on Cascadia Testbed

prominently featured two key components: a Wabco Smarttrac EBC and a Detroit Diesel CPC 3 EVO (Common Powertrain Controller) along with a Body Controller and a Cab Controller. However, their communication with external diagnostic tools and other networks was mediated by the Bosch gateway unit as shown in Fig. 6.

V. Attack Experiments

In this section, we describe the attack experiments conducted as part of our research. Each experiment is structured to include a research hypothesis, followed by detailed steps for hypothesis testing and an analysis of the results obtained. Finally, for each experiment, we discuss potential mitigation techniques that could be employed to counteract the identified vulnerabilities. It is important to note that all our tests were conducted from a black-box perspective, meaning that we did not have access to the source code or any runtime debug information of the systems being tested. This approach simulates the perspective of an external attacker with no insider knowledge, thereby ensuring the relevance of our findings to real-world scenarios.

A. Read Data by ID Overload Vulnerability

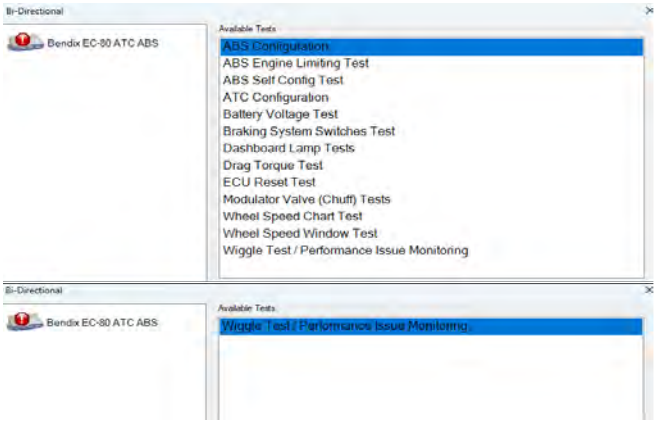
Our first attack is the Read Data by ID Overload attack, in which an attacker sends a large number of Read Data by ID requests to a specific ECU, overwhelming it and causing a denial of service condition.

1) Hypothesis: The ISO 14229-1 document specifies that upon receiving a Read Data by Identifier request, the ECU shall access the data elements of the records specified by the data identifier and transmit their value. We hypothesize that sending a high volume of Read Data by Identifier requests may overwhelm the ECU and prevent it from carrying out more critical tasks such as the transmission of periodic messages.

2) Testing: The attack was tested out on the local testbeds and also on the Freightliner Cascadia cab testbed. Chatterjee et al. [10] in their demonstration of targeted denial of service attacks on MHD networks, have already shown that messages with high J1939 priority `0x00` (0) can flood the CAN network by winning transmission arbitration. Thus targeted denial of service can only be achieved by sending low J1939 priority messages. In our experiments, we sent Read Data by Identifier (ID) requests with the lowest J1939 Priority `0x1C` (7) to the targeted ECU at varying intervals and observed any drop in periodic messages. By using a low priority, results are not conflated with arbitration mechanisms of CAN. The attack messages were sent at intervals of 0.1, 0.2, 0.3, 0.4, 0.5 and 0.6 milliseconds. Keep in mind, the messages may not end up on the network at these intervals due to CAN arbitration.

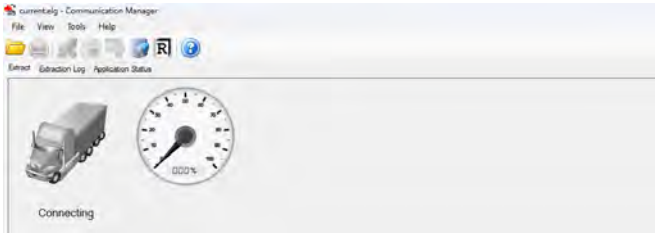
TABLE II: Read Data by ID effect on Normal Traffic

Attack Parameters			Average Message Count per ECU			
Testbed	Target ECU	Attack Message Interval (ms)	CPC		EBC	
			Count	% Decrease	Count	% Decrease
Testbed 1	EBC	0.1	206	0%	0	100%
		0.2	206	0%	0	100%
		0.3	206	0%	0	100%
		0.4	206	0%	57	24%
		0.5	206	0%	75	0%
		0.6	206	0%	75	0%
Testbed 2	EBC	0.1	132	0%	275	44%
		0.2	132	0%	350	29%
		0.3	132	0%	350	29%
		0.4	132	0%	492	0%
		0.5	132	0%	492	0%
		0.6	132	0%	492	0%
Testbed 3	CPC	0.1	0	100%	75	0%
		0.2	0	100%	75	0%
		0.3	0	100%	75	0%
		0.4	0	100%	75	0%
		0.5	110	47%	75	0%
		0.6	207	0%	75	0%
Testbed 4	CPC	0.1	0	100%	75	0%
		0.2	0	100%	75	0%
		0.3	0	100%	75	0%
		0.4	0	100%	75	0%
		0.5	0	100%	75	0%
		0.6	235	0%	75	0%

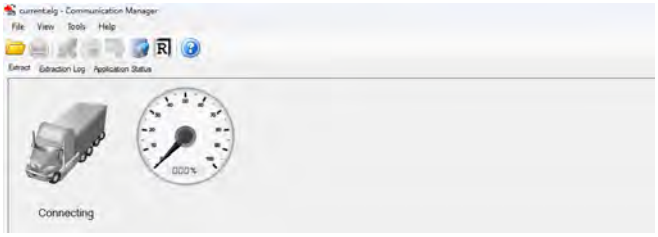


(a) Session Denial Results on Local Testbed 1

(b) Session Denial Results on Local Testbed 2



(c) Session Denial Results on Local Testbed 3



(d) Session Denial Results on Local Testbed 4

Fig. 9: Session Denial Attack on Local Testbeds

3) Results and Observations: As can observed from Fig. 7, there was a drop in normal periodic messages from the targeted ECU on each testbed when the Read Data by ID attack messages were sent at 0.3 ms intervals. However, the normal traffic from the other ECU on the testbed remained constant. These results and the impact of Read Data by ID messages at different intervals are further detailed in Table II. On the local testbed 1, where the Bendix EBC was the target ECU,

normal traffic started dropping when the attack messages were sent at 0.4 ms intervals, dropping to 0 at intervals lower than 0.4 ms. On the local testbed 2, where the Wabco EBC was the target ECU, normal traffic started dropping when the attack messages were sent at 0.3 ms intervals. On the local testbed 3, where the CPC3 was the target ECU, normal traffic started dropping when the attack messages were sent at 0.5 ms intervals, dropping to 0 at intervals lower than 0.5 ms. Finally, on the local testbed 4, where the CPC4 was the target ECU, normal traffic started dropping to 0 when attack messages were sent at intervals of 0.5 ms or lower. Our observations on the Freightliner Cascadia cab testbed further validated our findings. We recorded the number of messages on the network in 1 second during normal conditions, during diagnostics sessions, and during the Read Data by ID Overload attack on the EBC. As can be observed from Fig. 8, the number of messages on the network from the EBC increased during a diagnostic session than during normal conditions as would be expected. However, during the attack, the number of messages from the EBC dropped, while the messages from other ECUs on the internal network remained constant. This validates the Read Data by ID Overload as a targeted denial of service attack.

4) Possible Mitigation: A potential solution to defend against this kind of attack is for ECUs to ignore Read Data by ID request messages if they are received faster than a certain rate. Also, if these requests are processed by an interrupt service routine, the normal processes may be subverted, thus showing the decrease in the message traffic.

B. Session Denial Vulnerability

In our second experiment, we explore the *Session Denial Vulnerability*, hypothesizing that Diagnostic Session Control messages, coupled with Tester Present signals, could lead an ECU to neglect other valid session requests. This scenario could potentially block diagnostic tools and software from successfully connecting to the ECU.

1) Hypothesis: The ISO 14229-1 standard specifies that there shall always be exactly one diagnostic session active in an ECU. We hypothesize that by establishing a session with an ECU by sending Diagnostics Session Control messages followed by Tester Present signals to keep the session alive, the ECU may ignore other valid Diagnostic Session Control requests. This could prevent diagnostic tools and software from establishing a connection to an ECU.

2) Testing: The attack was carried out on both the local testbed and the Freightliner Cascadia cab testbed. A valid session was established with the target ECU using a Diagnostic Session Control Message from a spoofed source address, following which, the session was kept alive by sending Tester Present messages. During this time, attempts were made to establish a valid session using a diagnostic tool and the manufacturer's diagnostic software.

3) Results and Observations: As can observed from Fig. 9, when the attacker was in an active session with the target ECU, the diagnostic software could not establish a successful UDS session with the target ECU. Fig. 9a shows the diagnostic tests available during normal conditions and during the attack on local testbed 1. The Bendix A-COM diagnostic software was unable to provide all available tests during the attack. Similar results were seen on local testbed 2 as seen in Fig. 9b, where the Wabco Toolbox software was also unable to establish a diagnostic connection with the target ECU. Testbeds 3 and 4 showed the same results as seen in Fig. 9c and Fig. 9d. The DDEC Reports diagnostic software could not connect to the CPCs. The experiment was also tested on the Freightliner Cascadia testbed and yielded similar results. This is further discussed in Section V-D.

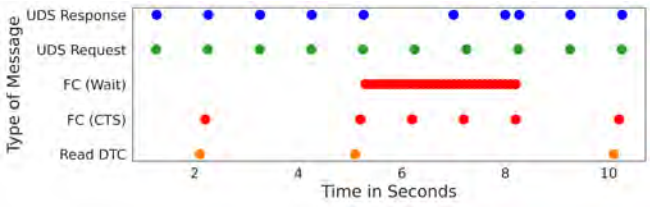
4) Possible Mitigation: Addressing the *Session Denial Vulnerability* identified in our experiments requires a layered security approach. One effective mitigation strategy is to implement a session request queue system that could allow the ECU to manage multiple session requests more efficiently, rather than being locked into a single session. To further enhance security, the introduction of source address validation for session requests can prevent unauthorized entities from establishing a session. This could involve authenticating diagnostic tools and software before allowing session initiation.

C. Diagnostics Jam Vulnerability

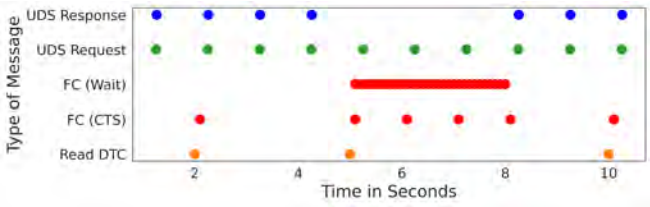
Our final experiment focuses on the *Diagnostics Jam Vulnerability*, where we test how sending a rapid mix of 'Wait' and 'Clear to Send' messages to an ECU can disrupt its normal operations and potentially lead to a service disruption.

1) Hypothesis: The ISO 15765-2 standard, commonly referred to as ISO-TP, facilitates communication and multipacket data transfer over the transport protocol between an external diagnostic tester and a vehicle's Electronic Control Unit (ECU). As per the protocol specifications, Flow Control (FC) frames are used to manage the transmission of multiframe messages, where 'Wait' frames indicate a pause in data transmission and 'Clear to Send' (CTS) frames signal the continuation of transmission. The protocol further specifies that an ECU shall keep track of the number of 'Wait' frames received in succession and terminate data transfer after it receives a certain number of 'Wait' FC frames in continuous succession specified by the manufacturer. However, this count is reset on receiving a CTS frame. Our hypothesis posits that a specific pattern of FC frames — repeatedly alternating between 'Wait' and 'CTS' within the maximum number of allowed 'Wait' frames — could exploit the ISO-TP's flow control mechanism. This may lead to a state where the ECU becomes overwhelmed and temporarily unable to process or respond to other diagnostic requests, effectively leading to a 'Diagnostics Jam' condition.

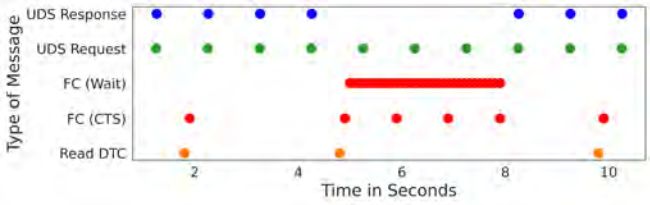
2) Testing: The attack was tested on both the local testbeds and the Freightliner Cascadia cab testbed. The testing process was set up as follows: A standard diagnostic tester consistently sent UDS requests in a routine manner. In parallel, our script was configured to request data for a Diagnostic Trouble Code (DTC) over the multipacket ISO-TP in a loop.



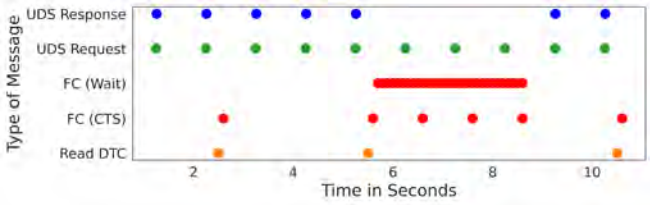
(a) Diagnostics Jam Attack Results on Local Testbed 1



(b) Diagnostics Jam Attack Results on Local Testbed 2

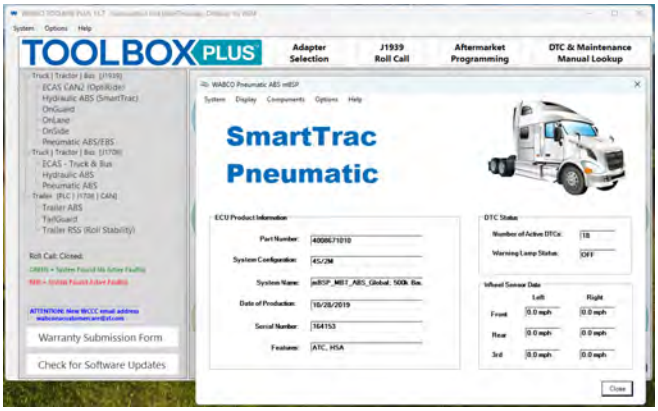


(c) Diagnostics Jam Results on Local Testbed 3

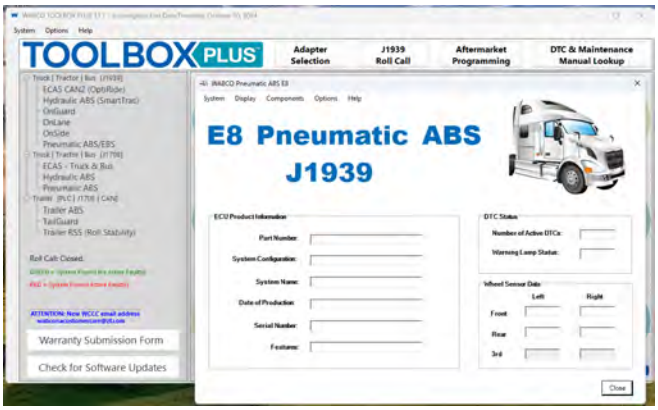


(d) Diagnostics Jam Attack Results on Local Testbed 4

Fig. 10: Diagnostics Jam Attack on Local Testbeds



(a) Diagnostics Software during Normal Conditions



(b) Diagnostics Software during Diagnostics Jam Attack

Fig. 11: Diagnostics Jam Attack on Freightliner Cascadia Cab Testbed

The script waited for the First Frame (FF) from the ECU. Once the FF was received, we sent a Flow Control (FC) message with a 'Clear to Send' (CTS) for data transfer. During the middle of this process, we strategically introduced a mix of 'CTS' and 'Wait' FC frames. Upon receiving a First Frame, we sent out a Flow Control (FC) message with a CTS for 1 packet followed by 10 FC frames with 'Wait' at intervals of 100 ms to prevent network flooding. This mixed sequence was maintained for a designated period before reverting back to the regular pattern of sending CTS frames only. The objective was to observe the interaction and potential impact of this mixed FC frame sequence on the ongoing UDS requests being handled by the ECU.

3) Results and Observations: As observed from Fig. 10, during our testing on local testbeds, the target Electronic Control Unit (ECU) displayed a standard behavior pattern under normal conditions, responding promptly to Unified Diagnostic Services (UDS) request messages during attempts to read Diagnostic Trouble Codes (DTC) over multi-packet data transfer. However, when the communication involved a mix of 'Clear to Send' (CTS) and 'Wait' frames, as per our hypothesized attack pattern, the response behavior of the ECU was absent.

During the attack, the target ECU reached a state where it became visibly overwhelmed. This was characterized by a significant delay in responding to UDS requests on local testbed 1, and a complete failure to respond on the other local testbeds. In essence, the ECU entered a 'Diagnostics Jam' condition, as hypothesized. The system's inability to process new diagnostic requests effectively rendered the diagnostic functionalities inoperative for the duration of the attack. Similar observations were made on the Freightliner Cascadia cab testbed. As can be seen in Fig. 11, during normal conditions the diagnostic software was able to gather relevant information from the EBC, however during the attack, it failed to gather the same information.

4) Possible Mitigation: To mitigate the vulnerabilities identified in the ISO 15765-2 standard, particularly against the 'Diagnostics Jam' attack, a multifaceted approach is recommended. This should include adjusting protocol timeouts

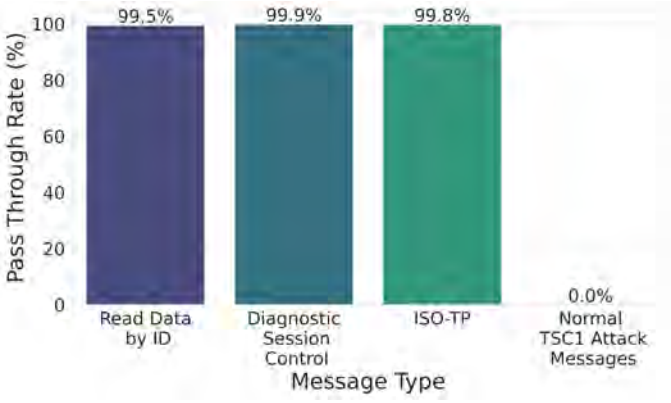


Fig. 12: Behavior of the Gateway ECU in Filtering Different Message Types

to counter abnormal flow control conditions, implementing anomaly detection algorithms to identify and react to suspicious patterns of Flow Control (FC) frames, and enhancing the ECU's processing capabilities to better handle high volumes of FC frames.

D. Gateway ECU Testing in the Freightliner Cascadia

Our investigation into the Freightliner Cascadia's network systems included a detailed analysis of the gateway Electronic Control Unit (ECU). As part of our experimental setup, we transmitted a variety of diagnostic messages and attack messages to observe the gateway ECU's response. Specifically, we sent messages from the diagnostic CAN network and tracked their passage to the internal ECU network, which is crucial for understanding the results shown in Fig. 12.

The outcomes from these tests reveal a significant disparity in the gateway ECU's treatment of message types. It consistently permitted almost every diagnostic message, including those related to 'Read Data by ID', 'Diagnostic Session Control', and 'ISO-TP'. However, it effectively obstructed all standard J1939 Torque/Speed Command 1 (TSC1) attack messages which were shown by Burakova et al. [7] in their attack scenario.

This evidence aligns with the observations made in the preceding sections on 'Read Data by ID Vulnerability', 'Session Denial Vulnerability', and 'Diagnostics Jam Vulnerability'. Each of these vulnerabilities was successfully executed on the Freightliner Cascadia's system, impacting its ECUs.

VI. Conclusion and Future Work

This paper presents three different scenarios where protocol vulnerabilities in unified diagnostics services (UDS) standards can be exploited to expose ECUs in Medium and Heavy Duty Vehicles to different types of attacks. First, two of the three scenarios demonstrate novel vulnerabilities in the ISO 14229 standard. The final scenario demonstrates a new attack case exploiting the ISO 15765 (Diagnostic Communication over CAN) specifications.

At its core, this paper helps in enhancing the existing threatscape of vehicle security for medium and heavy-duty vehicles. Security and functional testing should include these scenarios to catch potential logic issues in deployed components. A large part of the networking specifications remain unexplored for security loopholes and opportunities remain to investigate. Additionally, practical defense mechanisms to prevent these attacks are of keen interest.

Acknowledgment

This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) and Naval Information Warfare Center Pacific (NIWC Pacific) under Contract No. N66001-20-C-4021. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of DARPA or NIWC Pacific.

References

[1] Society of Automotive Engineers, "SAE J1939 Standards Collection." [Online]. Available: https://www.sae.org/standardsdev/groundvehicle/j1939a.htm

[2] International Organization for Standardization, "Information technology — Open Systems Interconnection — Basic Reference Model: The Basic Model," Standard ISO/IEC 7498-1:1994. [Online]. Available: https://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/02/02/20269.html

[3] Robert Bosch GmbH, "CAN Specification," Robert Bosch GmbH, Standard 2.0, 1991.

[4] M. Wolf, A. Weimerskirch, and C. Paar, "Security in Automotive Bus Systems," in *Proceedings of the Workshop on Embedded Security in Cars*. Bochum, Germany: Springer-Verlag, 2004, pp. 1–13.

[5] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, and T. Kohno, "Comprehensive Experimental Analyses of Automotive Attack Surfaces," in *USENIX Security Symposium*, vol. 4. San Francisco, CA, USA: USENIX Association, 2011, pp. 447–462.

[6] C. Miller and C. Valasek, "Remote Exploitation of an Unaltered Passenger Vehicle," in *Blackhat USA*. Las Vegas, NV, USA: Blackhat Press, 2015.

[7] Y. Burakova, B. Hass, L. Millar, and A. Weimerskirch, "Truck Hacking: An Experimental Analysis of the SAE J1939 Standard," in *Proceedings of the 10th USENIX Conference on Offensive Technologies*. Austin, TX, USA: USENIX Association, 2016, pp. 211–220.

[8] S. Mukherjee, H. Shirazi, I. Ray, J. Daily and R. Gamble, "Practical DoS Attacks on Embedded Networks in Commercial Vehicles," in *Proceedings of 12th International Conference on Information Systems Security*, 2016, pp. 23–42.

[9] P. Murvay and B. Groza, "Security Shortcomings and Countermeasures for the SAE J1939 Commercial Vehicle Bus Protocol," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 5, pp. 4325–4339, 2018.

[10] R. Chatterjee, S. Mukherjee, and J. Daily, "Exploiting transport protocol vulnerabilities in SAE J1939 networks," in *Proceedings of the Inaugural International Symposium on Vehicle Security & Privacy*. San Diego, CA, USA: Internet Society, 2023. [Online]. Available: https://www.ndss-symposium.org/wp-content/uploads/2023/02/vehiclesec2023-23053-paper.pdf

[11] International Organization for Standardization, "Road vehicles — Unified Diagnostic Services (UDS) — Part 1: Application Layer," Standard ISO 14229-1, 2020. [Online]. Available: https://www.iso.org/standard/72439.html

[12] International Organization for Standardization , "Road vehicles — Diagnostics Communication over Controller Area Networks (DoCAN) — Part 2: Transport and network layer services," Standard ISO 15765-2, 2016. [Online]. Available: https://www.iso.org/standard/66574.html

[13] J. Maag, C. Reding, and K. Howell, "Seed-key security exchange," https://www.engr.colostate.edu/~jdaily/presentations/2017%20Seed%20Key%20Exchange.pdf, 2017, presented at the 2017 Heavy Vehicle Cyber Security Workshop sponsored by the National Motor Freight Traffic Association, Inc.

[14] S. Kulandaivel, "Revisiting remote attack kill-chains on modern in-vehicle networks," PhD thesis, Carnegie Mellon University, 2021, available at: https://kilthub.cmu.edu/ndownloader/files/34106900.

[15] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, and S. Savage, "Experimental Security Analysis of a Modern Automobile," in *IEEE Symposium on Security and Privacy*. Oakland, CA, USA: IEEE, 2010, pp. 447–462.

[16] M. T. Campo, S. Mukherjee, and J. Daily, "Real-Time Network Defense of SAE J1939 Address Claim Attacks," *SAE International Journal of Commercial Vehicles*, vol. 14, no. 3, pp. 02–14–03–0026, Aug. 2021. [Online]. Available: https://www.sae.org/content/02-14-03-0026/