RESEARCH-ARTICLE

Denial of Service Vulnerabilities in Commercial Vehicles: Exploiting Diagnostic Protocol Flaws

CARSON GREEN, Colorado State University, Fort Collins, CO, United States

RIK CHATTERJEE, Colorado State University, Fort Collins, CO, United States

JEREMY S DAILY, Colorado State University, Fort Collins, CO, United States

Open Access Support provided by:

Colorado State University

Accepted: 04 August 2025
Revised: 08 May 2025
Received: 15 September 2024

Denial of Service Vulnerabilities in Commercial Vehicles: Exploiting Diagnostic Protocol Flaws

CARSON GREEN, RIK CHATTERJEE, and JEREMY DAILY, Colorado State University, USA

Commercial vehicles are a vital component of modern logistics and transportation, forming part of the critical infrastructure and representing safety-critical cyber-physical systems. Contemporary automotive operations are dominated by embedded computing systems that engage through standardized protocols, which constitute the infrastructure of vehicular communication networks. Within the commercial vehicle sector, these systems utilize high-level protocols that operate over the Controller Area Network (CAN) protocol for internal exchanges in medium and heavy-duty vehicles. The Unified Diagnostics Services (UDS) protocol, as described in International Standards Organization (ISO) 14229 (Unified Diagnostic Services - UDS) and ISO 15765 (Diagnostic Communication over CAN), plays a pivotal role by providing vital diagnostic capabilities. This research introduces four specific scenarios that expose deficiencies in the diagnostic protocol standards and how these can be manipulated to initiate attacks on in-vehicle computers within commercial vehicles, circumventing existing security frameworks. In the first three scenarios, we demonstrate three flaws within the ISO 14229 protocol standards. Following this, the fourth and final scenario elucidates a flaw unique to the ISO 15765 protocol standards.

For the purpose of demonstration, test setups incorporating actual Electronic Control Units (ECUs) linked to a CAN bus were employed. Further experiments were performed using a fully equipped cab assembly from a 2018 Freightliner Cascadia truck, set up as a testing environment. The experimental outcomes demonstrate how attacks targeting these specific protocols can undermine the integrity of individual ECUs, leading to denial of service. Additionally, within the Freightliner Cascadia configuration, a network architecture typical of contemporary vehicles was observed, featuring a gateway unit that isolates internal ECUs from diagnostic interfaces. Although this gateway is engineered to prevent conventional message injection and spoofing attacks, it permits all diagnostic communications. This selective permeability inadvertently introduces a susceptibility to diagnostic protocol flaws, highlighting an essential area for security improvements within commercial vehicle networks. These insights are vital for engineers and developers tasked with integrating the diagnostic protocols into their network subsystems, underscoring the urgency for improved security provisions.

CCS Concepts: • **Security and privacy** → **Denial-of-service attacks**; • **Networks** → *Cyber-physical networks*; *Transport protocols*.

Additional Key Words and Phrases: Unified Diagnostic Services, Commercial Vehicle Networks, Protocol Vulnerabilities, Denial of Service Attacks, Electronic Control Units

1 INTRODUCTION

Medium and heavy-duty (MHD) vehicles are integral to the critical infrastructure of the United States, serving key roles in freight transportation and the support of essential services such as emergency response. The increasing electrification of MHD vehicles has resulted in most mechanical functions being governed by embedded systems, commonly referred to as Electronic Control Units (ECUs). ECUs in MHD vehicles manage critical operations such as braking, steering, and power management while being interconnected through a bus topology network

Authors' address: Carson Green, Carson.Green@colostate.edu; Rik Chatterjee, Rik.Chatterjee@colostate.edu; Jeremy Daily, Jeremy.Daily@colostate.edu, Colorado State University, 711 Oval Drive, Fort Collins, Colorado, USA, 80521.

ACM 2378-9638/2025/8-ART
https://doi.org/10.1145/3760787

that handles mission-critical data. This integration makes MHD vehicles safety-critical cyber-physical systems, where any failure can directly affect human safety and the reliability of essential services. In MHD vehicles, communication within these networks is primarily governed by the Society of Automotive Engineers (SAE) J1939 standard [33]. This standard is organized in a layered structure, similar to the International Standards Organization/Open Systems Interconnection (ISO/OSI) model [16], which is widely used in traditional Information Technology (IT) networking. The lower-level physical layers of the SAE J1939 rely on the Controller Area Network (CAN) protocol [32] to enable data exchange within the vehicle.

Despite the robustness of CAN in automotive applications, its security properties, especially in MHD vehicles, require more detailed scrutiny. Multiple vulnerabilities, both remote and physical, have been shown to allow unauthorized control or disruption of vehicle functions [10, 11, 25, 26, 35]. Additionally, the SAE J1939 protocol, which underpins the cyber-physical operations of these vehicles, has been demonstrated to have its own security weaknesses [3, 8, 28, 29]. Research efforts have targeted different layers of the SAE J1939 standard, ranging from the application layer to the network management and data-link layers, revealing specific security vulnerabilities in each [2, 5, 6, 15, 19, 20, 27].

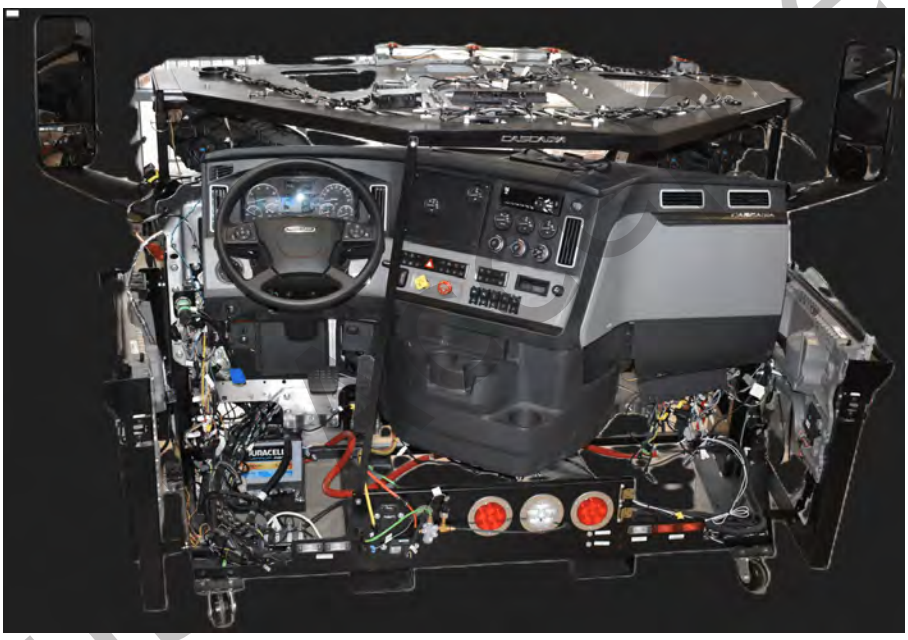

Fig. 1. Freightliner Cascadia Cab Testbed

One important area that remains relatively under-explored is the impact of diagnostic protocols, specifically the Unified Diagnostic Services (UDS) standards as defined in ISO 14229 [1, 18] and International Standards Organization (ISO) 15765 [17]. Diagnostic protocols are essential for ensuring the ongoing safety, efficiency, and operational health of vehicles. UDS offers a standardized framework for various diagnostic services, encompassing vehicle diagnostics, programming, and fault detection and resolution. Considering the role diagnostics play in the functioning of MHD vehicles, any vulnerabilities present within diagnostic standards could have serious ramifications, potentially undermining the performance, safety, and overall integrity of vehicular networks.

Although significant research has been done to identify and exploit weaknesses in the authentication processes used in diagnostic protocols [22, 24, 26], there is a notable lack of focus on vulnerabilities inherent within

Start of Frame	Arbitration Field	Control Field	Data Field	CRC Field	ACK Field	End of Frame
(1 bit)	(29 bits)	(6 bits)	(0-8 bytes)	(16 bits)	(2 bits)	(7 bits)

Fig. 2. CAN Message Structure

the diagnostic protocols themselves. This paper aims to address that gap by examining deeper, less obvious flaws within these protocols through a black-box approach. Our research turns its attention to exploring the vulnerabilities embedded in the UDS standards specific to MHD vehicles. This focus diverges from earlier works, which primarily dealt with issues related to seed-key exchanges in UDS for passenger vehicles. Instead, we examine the underlying weaknesses in the UDS protocol itself, identifying potential attack vectors that do not rely on vulnerabilities specific to or requiring authentication.

Additionally, our study highlights an important discovery regarding the network architecture of the Freightliner Cascadia cab, as depicted in Fig. 1, which can be found in most modern MHD vehicles. This testbed replicates most of the wiring and electronics present in the cab of a standard Freightliner Cascadia heavy-duty truck. A key observation was that the separation of internal ECUs from diagnostic interfaces through a gateway unit created a distinct scenario: while the gateway blocks typical message injection and spoofing attacks, it allows all diagnostic traffic to pass through unfiltered. This is an important finding, as it implies that gateway-permitted diagnostic messages could be used to exploit weaknesses in the diagnostic protocols, opening a new attack surface in MHD vehicle networks. As a result, our research emphasizes the need to reconsider security strategies with a specific focus on safeguarding diagnostic communications and the role of gateway configurations in MHD vehicles.

This paper seeks to expand the current understanding of the threat landscape for in-vehicle network applications in MHD vehicles, stressing the importance of implementing comprehensive security measures to address emerging cyber threats. The remainder of this paper is structured as follows: Section 2 provides an overview of the relevant protocol standards, Section 3 describes the threat model considered for this research, Section 4 surveys related literature in this area, Section 5 details the test setup, Section 6 describes the experiments conducted and key findings, and Section 7 offers concluding thoughts and suggestions for future research.

2 BACKGROUND

The communication infrastructure within MHD vehicles depends on several key protocols. These include SAE J1939 over a CAN physical, ISO 14229 (Unified Diagnostic Services - UDS), and ISO 15765 (Diagnostic Communication over CAN). Each of these protocols plays a critical role, ranging from overseeing routine vehicle operations to managing diagnostics and ensuring safety. In this section, we will examine each protocol in greater detail to better understand its role in facilitating communication within MHD vehicles.

2.1 CAN

CAN is a serial communication protocol used for in-vehicle networking. Each CAN message consists of an identifier, control field, data field, CRC field, acknowledgment, and end-of-frame segments shown in Fig. 2. In MHD vehicles, the extended frame format is commonly used, which features a 29-bit identifier. This identifier determines the message's priority during arbitration and allows for differentiation between messages. The standard data frame can carry up to 8 bytes of payload. The control field specifies the data length, while the CRC field enables error detection. CAN operates using a multi-master architecture, where any node can initiate transmission if the bus is idle. Arbitration is non-destructive, ensuring that the highest-priority message is

Priority	PDU Format	PDU Specific	Source Address	Data Field
(3 bits)	(8 bits)	(8 bits)	(8 bits)	(0 - 1785 bytes)

Fig. 3. SAE J1939 PDU Structure

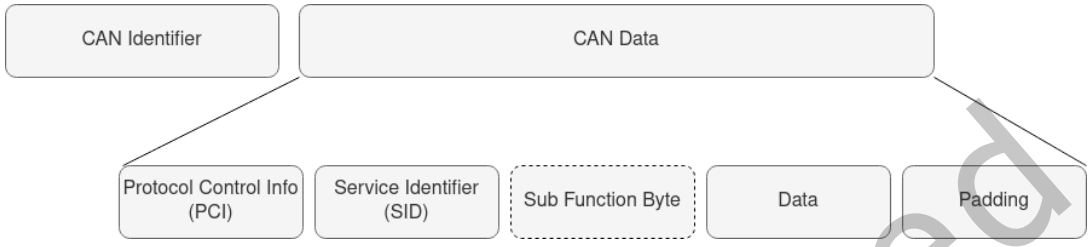

Fig. 4. UDS Message Format in Medium and Heavy Duty Vehicles

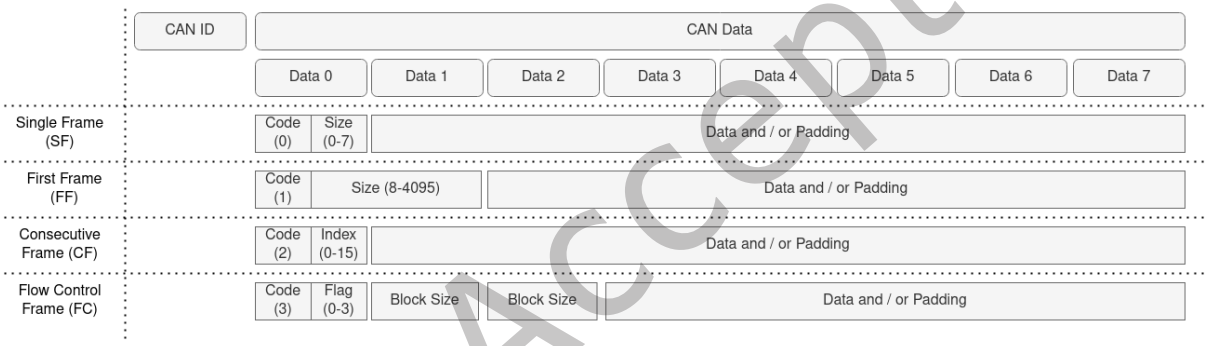

Fig. 5. ISO 15765 Message Format

transmitted without collision. Error-handling features include automatic retransmission, error counters, and fault confinement states that allow nodes to enter passive or bus-off modes when error thresholds are exceeded.

2.2 SAE J1939

In medium and heavy-duty vehicles, in-vehicle communication is primarily governed by the SAE J1939 standards, which operate over the physical controller area network. SAE J1939 messages encapsulate various operational parameters, such as engine speed, vehicle speed, and switch statuses. These parameters are grouped into logical sets known as Parameter Groups (PGs). Each PG is assigned a unique identifier, referred to as a Parameter Group Number (PGN), which is embedded within the message. Data transmitted through J1939 messages is contained in a J1939 Protocol Data Unit (PDU) illustrated in Fig. 3. The PDU includes a source address (SA) that identifies the sender, a destination address (DA) for the recipient, the message priority, the PGN, and up to 1785 bytes of data. For messages with 8 bytes or fewer, the priority, PGN, SA, and DA are embedded in the identifier (ID) field of the CAN frame. When PDUs exceed 8 bytes, a transport protocol (TP) is employed, with the PGN being located in the final 3 bytes of the TP Connection Management (CM) message data. For diagnostic messages, SAE J1939 designates a specific PGN 55808 (`0x0DA00`).

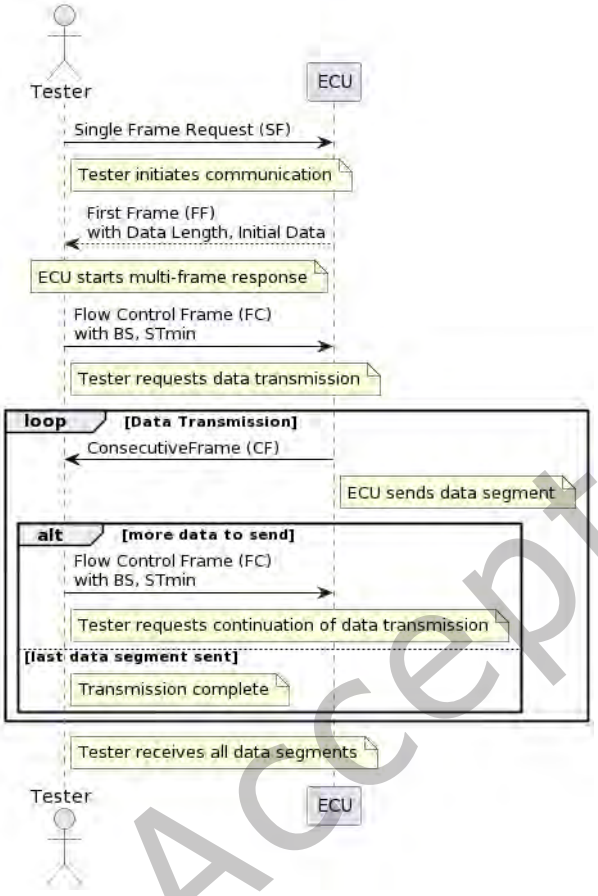

Fig. 6. Logical Point-to-Point Multiframe Data Transfer using ISO-TP

2.3 ISO 14229: Unified Diagnostic Services

ISO 14229, commonly known as UDS, is a fundamental application-layer protocol for automotive diagnostics, enabling communication between a vehicle's ECUs and external diagnostic tools. UDS revolves around several critical components that organize the diagnostic communication flow, as depicted in Fig. 4. One of the main elements is the Protocol Control Info (PCI), which identifies the type of UDS message, its size, and other relevant parameters. Alongside the PCI is the Service Identifier (SID), which plays a crucial role in specifying the diagnostic service or function being requested or executed. In many UDS messages, there is also a Sub-function field, which provides additional instructions or details related to the requested diagnostic service. The Data segment then conveys the specific information or commands relevant to the service request. This systematic structure allows for a broad spectrum of diagnostic operations, including reading or writing data, conducting tests, and retrieving information from ECUs or the vehicle.

Table 1 illustrates common SIDs used in UDS, along with the corresponding request codes, positive response codes, and negative response codes.

Table 1. 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

2.4 ISO 15765: Diagnostic Communication over CAN

ISO 15765 is a key automotive standard that facilitates session-layer communication over CAN, particularly for diagnostic purposes. As illustrated in Fig. 6, this protocol is essential for handling the transmission of data packets that exceed the size limit of a single CAN frame, a crucial capability for tasks like in-depth diagnostics and ECU programming. ISO 15765 transmits data through several different frame types, as outlined in Fig. 5, each serving a unique purpose and structure:

(1) **Single Frame (SF)**: Designed for data payloads up to 7 bytes. It starts with a 0 in the first nibble, followed by the data length in the subsequent nibble (half-byte). The remaining bytes carry the actual data content.
(2) **First Frame (FF)**: Initiates the transmission of multi-frame data for payloads exceeding 7 bytes. The first nibble begins with 1, and the next 3 nibbles (12 bits) indicate the total data length. The rest of the frame contains the first portion of the data.
(3) **Consecutive Frame (CF)**: Used to send subsequent chunks of data following the first frame. It starts with a 2 in the first nibble, followed by a frame number in the second nibble, which increments with each consecutive frame, ensuring proper sequencing.
(4) **Flow Control Frame (FC)**: Regulates the flow of data in a multi-frame message. The frame begins with 3 in the first nibble, and the next nibble indicates the flow control status: 0 for Continue To Send (CTS), 1 for Wait, and 2 for Overflow/Abort. CTS signals the sender to continue transmission, Wait requests a pause, and Overflow/Abort informs the sender that the receiver cannot handle more data. Additionally, this frame defines how many frames can be sent at a time and the delay between them.

These frame types collectively support the transfer of large data packets over CAN. Single frames manage smaller packets, while the combination of first, consecutive, and flow control frames ensures efficient and reliable transmission of larger data sets.

3 THREAT MODEL

This section outlines the attacker threat model relevant to the research, detailing the potential entry points, attacker assumptions, and the capabilities required to exploit diagnostic protocol vulnerabilities within the vehicle's network.

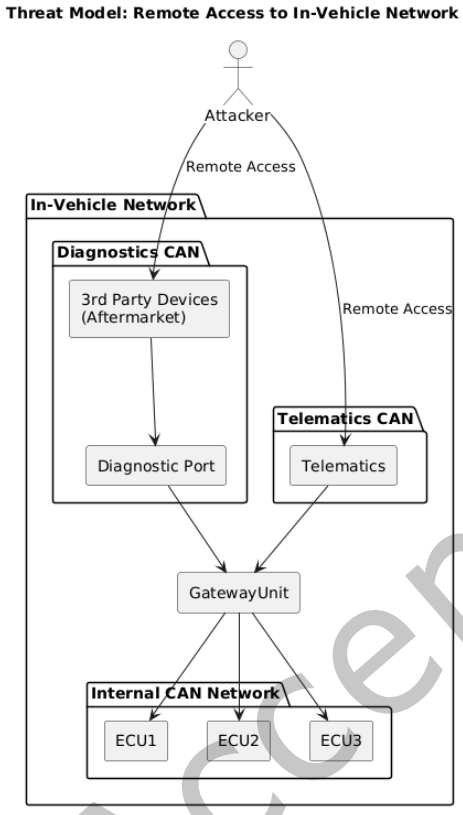

Fig. 7. Attacker Threat Model

3.1 Assumptions

In this threat model, we assume the attacker has remote access to the vehicle's network. This access can be achieved through various entry points, including telematics systems or third-party aftermarket devices connected to the diagnostic port. As shown in Fig. 7, telematics connect to the vehicle's gateway via the Telematics CAN, while third-party devices interface with the vehicle through the diagnostic port on the Diagnostics CAN. Both paths provide access to the gateway, which separates the internal ECUs on the Internal CAN network.

The attacker does not require insider privileges but instead exploits poorly secured access points within the vehicle network. This may involve compromising aftermarket devices, as demonstrated by Jepson et al. [19], exploiting a diagnostics tool connected to the exposed On-Board Diagnostics port as shown by Kumar et al. [23], or remotely compromising an insecure telematics unit, as shown by Miller et al. [26]. The attacker is knowledgeable of the diagnostic protocol, which is publicly available and widely used across the automotive industry, allowing them to craft malicious messages targeting critical ECUs.

3.2 Attacker Capabilities

The attacker is assumed to have remote access to the vehicle's network through vulnerabilities in telematics systems, third-party devices, or diagnostic ports. Once access is obtained, the attacker can send spoofed diagnostic

messages, bypassing security mechanisms and interacting directly with the vehicle's ECUs. These capabilities include the ability to manipulate ECU behavior, initiate unauthorized diagnostic sessions, and disrupt vehicle operations.

Furthermore, the attacker can exploit weaknesses in wireless interfaces such as Wi-Fi, Bluetooth, or cellular, as well as unsecured API connections. Vulnerabilities in outdated diagnostic computers used by technicians, which run old or unpatched software, can also be leveraged to inject malicious payloads that modify or manipulate diagnostic communications. The attacker's primary method of attack is through the injection of spoofed diagnostic messages to gain control of critical vehicle systems.

3.2.1 Potential Attack Vectors. The following attack vectors are considered in this threat model, assuming the attacker has gained access to the in-vehicle network:

- **Exploitation of Third-Party Devices:** Malicious or compromised aftermarket devices connected to the diagnostic port can be used to inject unauthorized diagnostic messages into the vehicle network, gaining access to critical systems and bypassing gateway protections.
- **Session Hijacking:** The attacker can intercept and take control of active diagnostic sessions, using crafted messages to manipulate ECU functions, downgrade sessions, or disrupt critical operations such as software updates.
- **Denial of Service (DoS) through Diagnostic Message Flooding:** By flooding the network with diagnostic requests, such as repeated Tester Present messages, the attacker can overwhelm ECUs, preventing legitimate diagnostic tools from accessing the vehicle's systems.
- **Gateway Exploitation:** By exploiting vulnerabilities in the vehicle's gateway, the attacker can bypass security filters that control access between external networks and internal ECUs, enabling unauthorized access to sensitive vehicle functions.
- **Remote Code Execution and Control:** Vulnerabilities in telematics systems, third-party diagnostic devices, and unsecured network interfaces (Wi-Fi, Bluetooth, cellular) can be exploited to execute remote code or send crafted UDS commands. This could lead to remote control of vehicle functions, as demonstrated in past research by Miller et al. for telematics and Jake et al. for third-party devices.
- **Vulnerable Diagnostic Computers:** Outdated or poorly secured diagnostic tools used by technicians can be compromised shown by Kumar et el. [23], allowing attackers to inject malicious payloads or manipulate communication between diagnostic devices and the vehicle's ECUs.

3.3 Consequences

Exploiting these vulnerabilities could result in significant consequences, such as disabling vehicle diagnostics, interrupting software updates, or manipulating critical vehicle functions like braking or acceleration. DoS attacks could prevent essential maintenance or recovery operations, which is particularly hazardous for fleet and heavy-duty commercial vehicles, where operational downtime incurs substantial costs.

4 RELATED WORK

The security concerns surrounding automotive protocols, while crucial, have traditionally received limited attention in research. However, recent investigations have started to expose a range of vulnerabilities within these protocols.

Kosher et al. [21] highlighted weaknesses in seed-key exchanges between ECUs in passenger vehicles, revealing that the widely adopted 8 or 16-bit seed-key pairs are vulnerable to brute-force attacks. This discovery raises serious questions about how easily authenticated security sessions can be breached, potentially allowing unauthorized access to critical ECU operations. Building on this, Miller et al. demonstrated the possibility of bypassing security authentication for ECUs in passenger cars. By reverse engineering the firmware of diagnostic software in both

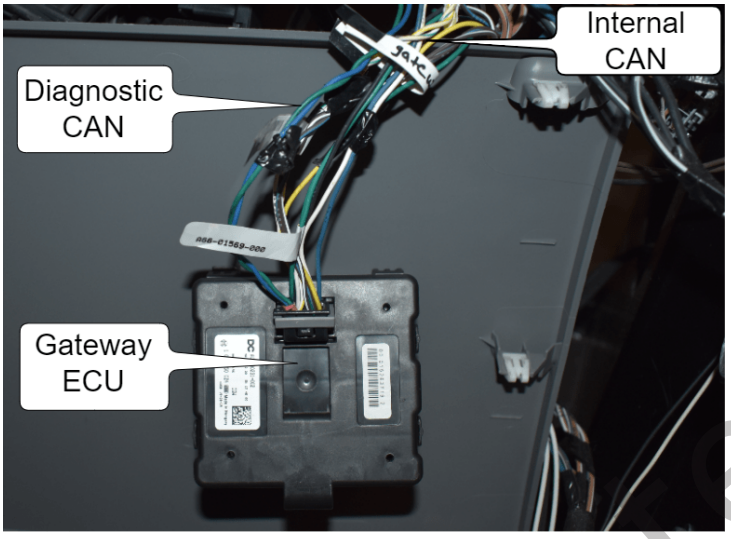

Fig. 8. Gateway Unit in Cascadia Testbed

a Ford Escape and a Toyota Prius, they were able to uncover the algorithm used to derive keys from seeds. Their work showcased several real-world attacks, such as tampering with brakes, lights, and engine control, underscoring significant security shortcomings.

Burakova et al. identified weaknesses within the application layer of the SAE J1939 Protocol. Their research illustrated how continuous control over a truck's engine could be achieved by leveraging specific J1939 messages. Moreover, they demonstrated how critical functionalities, such as engine braking and accelerator input, could be compromised, underscoring the risks involved in commercial vehicle operations. Mukherjee et al. and Chatterjee et al. [7–9, 28] concentrated on data-link layer protocols, exposing how rapid request messages to an ECU could overwhelm its processing capacity. Additionally, they found that illegitimately sustained connections to an ECU could block legitimate connections, creating a subtle yet effective denial-of-service condition. In the area of network management, Murvay et al. [29] demonstrated how flooding the network with specific address claim messages could render ECUs non-operational. They also showed that multi-packet data transfers could be interrupted by abrupt terminations, resulting in a denial-of-service attack. Campo et al. confirmed the address claim vulnerability and proposed a real-time mitigation solution [4]. Ghatak et al. and Olufowobi et al. explored the feasibility of constructing secure, fault-tolerant vehicular networks [12–14, 31]. Nyambe et al. explored software bill of materials management of embedded devices in vehicular networking [30].

Maag et al. [24] expanded the understanding of cybersecurity issues in seed-key exchanges between ECUs and vehicle diagnostic adapters (VDAs) in MHD vehicles. Their work highlighted a linear relationship in seed-key pairs, making it possible to predict these pairs in 16-bit configurations. Lastly, Kulandaivel et al. [22] identified several vulnerabilities in UDS implementations within passenger cars. His research centered on securing access to ECUs, revealing that the seeds used in challenge-response pairs were not entirely random and could be influenced by ECU uptime. This insight into predictable seed generation further accentuates the security vulnerabilities in automotive protocols.

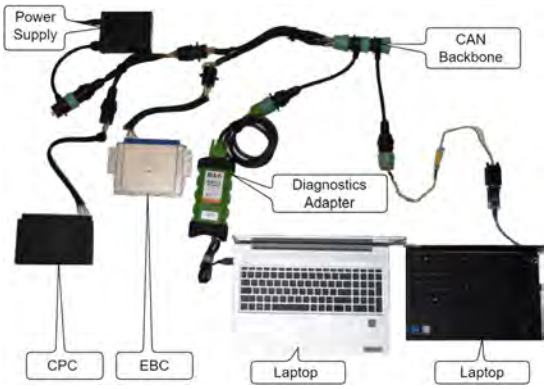

Fig. 9. Benchtop Testbed 1

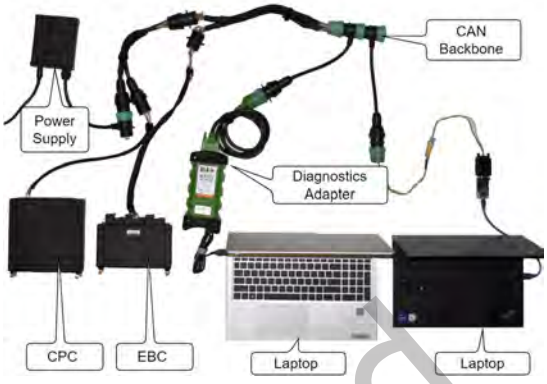

Fig. 10. Benchtop Testbed 2

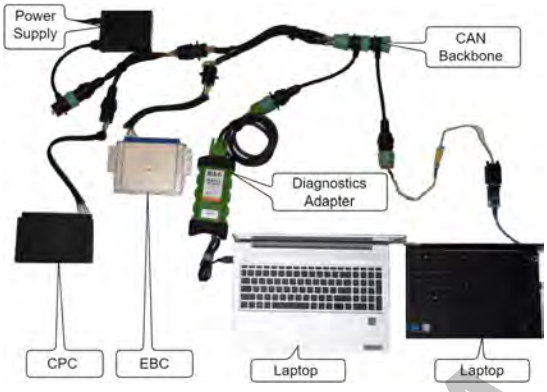

Fig. 11. Benchtop Testbed 3

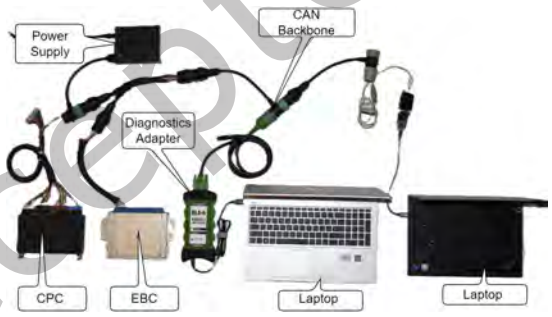

Fig. 12. Benchtop Testbed 4

5 EXPERIMENTAL TESTING SETUP

To ensure accurate and consistent results, we conducted our experiments using multiple configurations on both a local bench testbed and a comprehensive assembly of a 2018 Freightliner Cascadia cab. Each testbed featured a minimum of one ECU that communicated using the UDS protocol. This section outlines the various configurations of the local testbeds and the Freightliner Cascadia testbed, detailing the components used, the baud rate of the CAN bus, and the assigned addresses of each ECU.

5.1 Bench Testbed Configurations

The local bench testbeds were organized into four distinct configurations, each designed to test different vulnerabilities within UDS communication. Each setup contained a target ECU, typically an Electronic Brake Controller (EBC), and a control ECU, such as a Common Powertrain Controller (CPC). A typical vehicle has only one CPC for engine and transmission control, and one brake ECU, which means each testbed is fully representative of the communication behavior between the two ECUs present without added noise from other ECUs such as a cab controller. The setups varied based on the specific components and the speed of communication over the CAN bus. The SAE J1939 protocol defines an ECU SA based upon function, such as the CPC receiving an address of 0 (0x00) whereas an EBC receives an address of 11 (0x0B), such that in a full vehicle each ECU would have one or

more unique addresses. Additionally, the data rate for each testbed is not manually set by the authors, but by the ECU manufacturer for compliance with other ECU data rates. Each vehicle architecture may vary dependent on manufacturer such as having multiple CAN networks, yet the testbed setups below do not suggest or represent a specific architecture, only function of communication for vulnerability assessment.

Testbed 1: This configuration consisted of a Bendix EC-80 EBC paired with a Detroit Diesel CPC 3, both communicating over a 250 kbps CAN bus. The CPC 3 was assigned a Controller Application (CA) address of 0 (0x00), while the EBC was assigned the address 11 (0x0B). In this setup, the Bendix EC-80 EBC was the target ECU for testing vulnerabilities, whereas the Detroit Diesel CPC 3 was assigned as the control ECU.

Testbed 2: In this configuration, a Wabco Smarttrac system was paired with a Detroit Diesel CPC 3 EVO, operating over a faster 500 kbps CAN bus. The addressing scheme remained the same, with the CPC 3 EVO assigned an address of 0 (0x00) and the EBC assigned an address of 11 (0x0B). The Wabco Smarttrac EBC was the target ECU for this test, while the Detroit Diesel CPC 3 EVO acted as the control ECU.

Testbed 3: Similar to Testbed 1, this configuration included a Bendix EC-80 EBC and a Detroit Diesel CPC 3, operating on a 250 kbps CAN bus. The addressing scheme was the same, with the CPC 3 having an address of 0 (0x00) and the EBC assigned the address 11 (0x0B). The CPC 3 was the target ECU for this testbed, with the Bendix EC-80 EBC serving as the control ECU.

Testbed 4: This final bench setup featured a Bendix EC-80 EBC paired with a Detroit Diesel CPC 4, again operating on a 250 kbps CAN bus. The addressing scheme was similar to Testbed 1, with the CPC 4 having an address of 0 (0x00) and the EBC an address of 11 (0x0B). The CPC 4 was the target ECU for this configuration, as the EBC was responsible for being the control ECU.

Each of these testbeds featured a Linux laptop running SocketCAN and the 'can-utils' software for capturing and transmitting CAN messages. These laptops also functioned as the attack points for initiating different types of UDS-based vulnerabilities. Power supplies were included in each testbed setup to simulate the vehicle's electrical environment. A separate laptop equipped with a Noregon DLA, acting as the RP1210-compliant VDA, was utilized to interface with the target ECUs within each testbed.

5.2 Freightliner Cascadia Testbed

The second set of experiments was conducted on a 2018 Freightliner Cascadia cab, which served as a comprehensive real-world testbed. This testbed primarily consisted of a Wabco Smarttrac EBC and a Detroit Diesel CPC 3 EVO, along with other critical components like a Body Controller and Cab Controller. Communication between these components and the external diagnostic tools was negotiated from a Bosch gateway unit. The setup allowed for exploration of the potential vulnerabilities in UDS communication between internal and external networks, as shown in Fig. 8.

6 ATTACK EXPERIMENTS

This section elaborates upon the attack experiments carried out during our research. Each of the experiments is organized to include a research hypothesis, followed by detailed procedures for testing each hypothesis and ending with an analysis of the resulting data. Additionally, for each experiment, we explore conceivable mitigation approaches that could be applied to address the identified vulnerabilities. We emphasize the fact that our experiments were guided from a black-box context, meaning there was no access to run-time debug information or source code of the systems under evaluation. This approach mirrors the viewpoint of an external malicious actor with no insider knowledge, therefore enhancing the real-world applicability of the findings presented. All data generated from our experiments are hosted in a public repository [34].

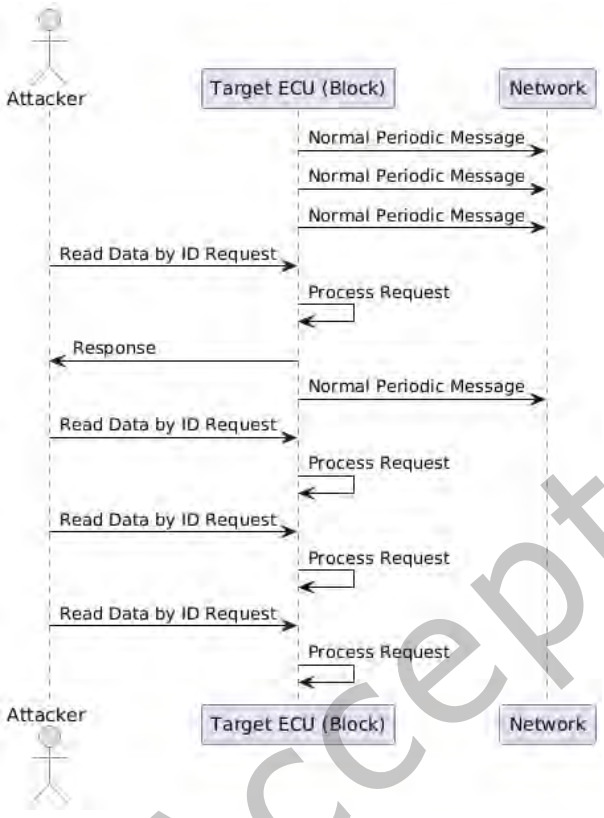

Fig. 13. Read Data by ID Overload Vulnerability Hypothesis

6.1 Read Data by ID Overload Vulnerability

Our first attack is named the Read Data by ID Overload Attack, which consists of an attacker sending a large number of Read Data by ID requests to a targeted ECU, in which a denial of service scenario is created by disabling ECU functionality and response.

6.1.1 Hypothesis. The ISO 14229-1 document states upon reception of a Read Data by Identifier request, the ECU is required to retrieve data elements corresponding to the specified data identifier and transmit their associated values. As outlined in Fig. 13, we hypothesize that transmitting a large volume of the Read Data by Identifier requests could overload the target ECU forcing it to look up data to serve the request, hindering its ability to perform essential functions, such as transmitting essential messages periodically that are required by other ECUs and sensors to maintaining vehicle functions.

6.1.2 Testing. The Read Data by Identifier attack was conducted on both the local testbeds as well as on the Freightliner Cascadia testbed. Chatterjee et al. [8] have previously demonstrated that denial of service attacks on MHD vehicle networks can exploit high-priority J1939 messages `0x00` (0) to flood the CAN bus network during transmission through consistent winning of arbitration. Therefore, to achieve a targeted denial of service in our trials, we utilized low-priority J1939 messages. Specifically, we transmitted Read Data by Identifier requests with the low priority of `0x1C` (7) to the target ECU at various intervals and monitored the network for any reduction

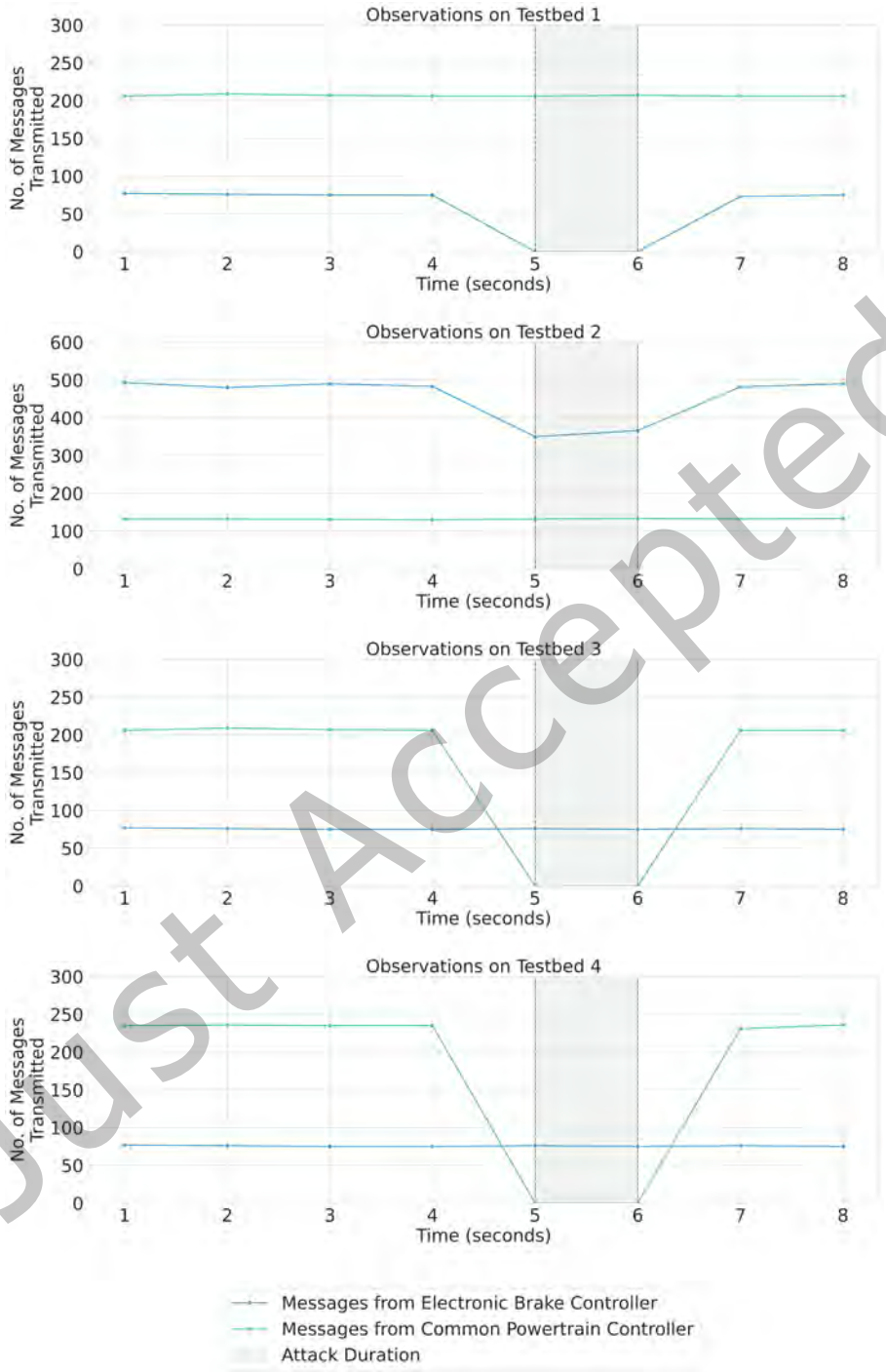

Fig. 14. Read Data by ID Overload Attack at 0.3 ms Interval Attack Messages

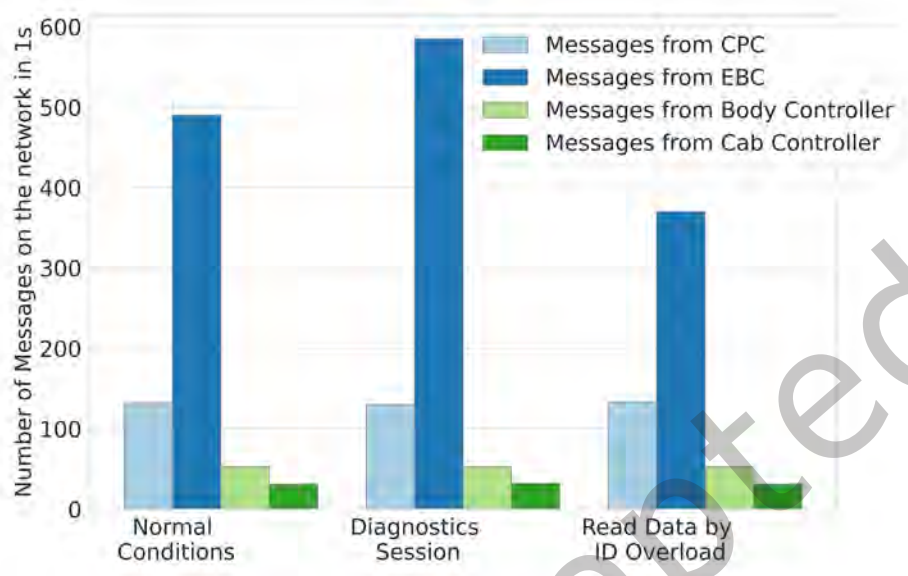

Fig. 15. Read Data by ID on Cascadia Testbed

in periodic message transmission. Taking advantage of low priority in our transmissions ensured the results were not confounded by CAN arbitration mechanisms. The attack messages were broadcast at intervals from 0.1-0.6 milliseconds with a step size of 0.1 milliseconds, though it is important to note that CAN arbitration may cause messages to have differing interval values.

6.1.3 Results and Observations. As illustrated in Fig. 14, there was a noticeable decline in normal periodic messages from the targeted ECU across all testbeds when the Read Data by ID attack messages were sent at intervals of 0.3ms. However, the normal traffic from the control ECU on the testbed remained stable. The detailed results, along with the effects of varying intervals for the attack messages are outlined in Table 2. Local testbed 1, containing the Bendix EBC as the target ECU, began to demonstrate a decrease in normal traffic when the attack messages were transmitted at 0.4ms intervals, eventually dropping to 0 messages at intervals below 0.4 ms. For local testbed 2 containing the Wabco EBC target ECU, traffic reduction occurred when the attack messages were transmitted at intervals of 0.3 ms and below. Local testbed 3, consisting of the CPC3 target ECU, showed that all intervals at 0.5 and lower decreased the count of periodic messages, with the count being 0 at 0.4 ms and below. Lastly, on local testbed 4 with the target ECU being the CPC4, normal traffic ceased entirely when attack messages were sent at intervals of 0.5 ms or less. Table 2 demonstrates that as injection frequency increases, the effect of the attack becomes more successful as correlated between the ”Interval” and ”Average Message Count” columns.

The effect of the Read Data by ID attack on the Freightliner Cascadia cab testbed further corroborated our findings. We measured the count of messages on the network over a 1-second period during normal conditions, during diagnostic sessions, and throughout the Read Data by ID Overload attack on the testbed’s EBC. As depicted in Fig. 15, the number of messages from the EBC increased during a diagnostics session compared to normal conditions, as expected. However, under attack, the EBC demonstrated a decrease in the count of messages while

the message traffic from other ECUs on the internal network remained unchanged. This confirms that the Read Data by ID Overload constitutes a targeted denial of service attack.

6.1.4 Possible Mitigation. A possible defense mechanism against a targeted denial of service, such as the Read Data by ID Overload attack, is to have an ECU disregard the specific Read Data by ID request messages that are received at a rate exceeding a predefined threshold. Additionally, if such requests are handled by an interrupt service routine, the routine may disrupt normal processes which could explain the observed reduction in message traffic. This suggests that rate-limiting or prioritizing critical processes to functionality could help mitigate the impact of such attacks.

6.2 Session Denial Vulnerability

In our second experiment, we investigate the *Session Denial Vulnerability*, hypothesizing that transmitting spoofed Diagnostic Session Control messages combined with Tester Present signals could cause an ECU to disregard other legitimate session requests over the network. This condition may result in diagnostic tools and software being unable to establish a successful connection with an ECU, effectively preventing access to critical diagnostic functions.

6.2.1 Hypothesis. The ISO 14229-1 standard mandates that only one diagnostic session may be active at any given time with an ECU. The ISO 14299-2 standard states that continuous Tester Present messages every 5000 ms are required to keep an active session ongoing. We hypothesize that by initiating a spoofed session with an ECU through Diagnostic Session Control messages, followed by Tester Present signals to maintain the session, the ECU may be locked into a session and overlook other valid Diagnostic Session Control requests. This behavior is depicted in Fig. 16, which could effectively block diagnostic tools and software from connecting to the ECU, potentially hindering essential diagnostic operations.

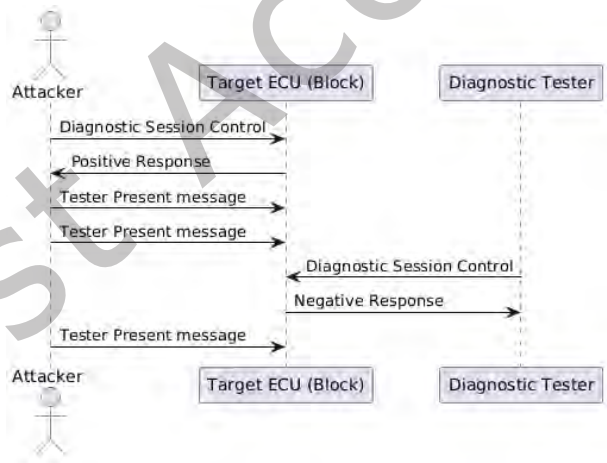

Fig. 16. Session Denial Vulnerability Hypothesis

6.2.2 Testing. The Session Denial attack was executed on both the local testbeds and the Freightliner Cascadia cab testbed. First, a valid session was initiated with the target ECU which utilized a Diagnostic Session Control message from a spoofed source address. The session was then maintained by periodically transmitting Tester Present frames. While the session was kept alive, attempts were made to establish a legitimate session with

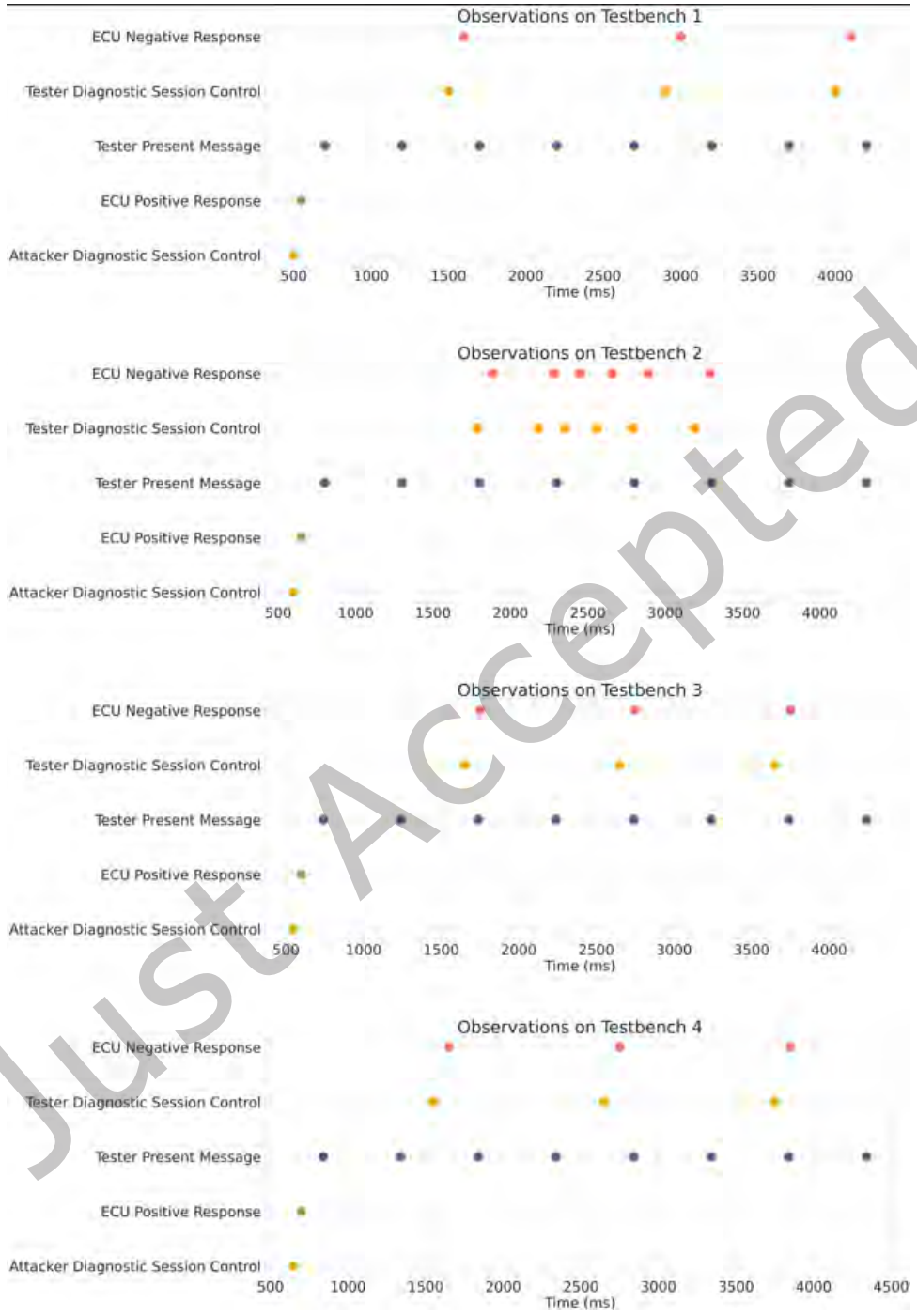

Fig. 17. Session Denial Vulnerability Demonstration

Table 2. 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%

the ECU using a diagnostic tool and the manufacturer's diagnostic software, testing the ECU's ability to handle multiple diagnostic session requests.

6.2.3 Results and Observations. As shown in Fig. 17, when the attacker achieved and maintained an active session with the target ECU, the diagnostics software was unable to establish a successful UDS session with the target ECU. The graph for local testbed 1 illustrates an unsuccessful attempt to initiate a legitimate diagnostic session during the attack. Due to this, the Bendix A-COM diagnostic software was unable to connect with its ECU counterpart. A similar outcome was observed on local testbed 2, where the target ECU and the Wabco Toolbox diagnostic software were unable to establish a diagnostic connection. Both testbeds 3 and 4 exhibited the same pattern, where the DDEC Reports diagnostic software tool was unable to establish a session with the CPC ECUs. The experiment was also conducted on the Freightliner Cascadia cab testbed, presenting analogous results, which is shown Fig. 18.

6.2.4 Possible Mitigation. Mitigating the *Session Denial Vulnerability* detected in our experiments necessitates a layered security approach. An effective strategy is to utilize a session request queue system, therefore enabling

Fig. 18. Session Denial Vulnerability Demonstration on Cascadia Testbench

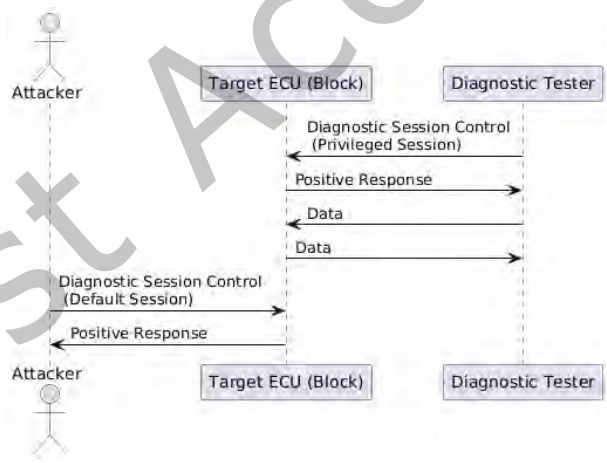

Fig. 19. Session Downgrade Vulnerability Hypothesis

the ECU to handle multiple session requests simultaneously, rather than being restricted to a single session. Additionally, security may be further enhanced by introducing source address validation for session requests, restricting unauthorized entities from initiating sessions. This can be achieved by authenticating diagnostic tools and software before granting session access, thereby ensuring only authorized connections are allowed.

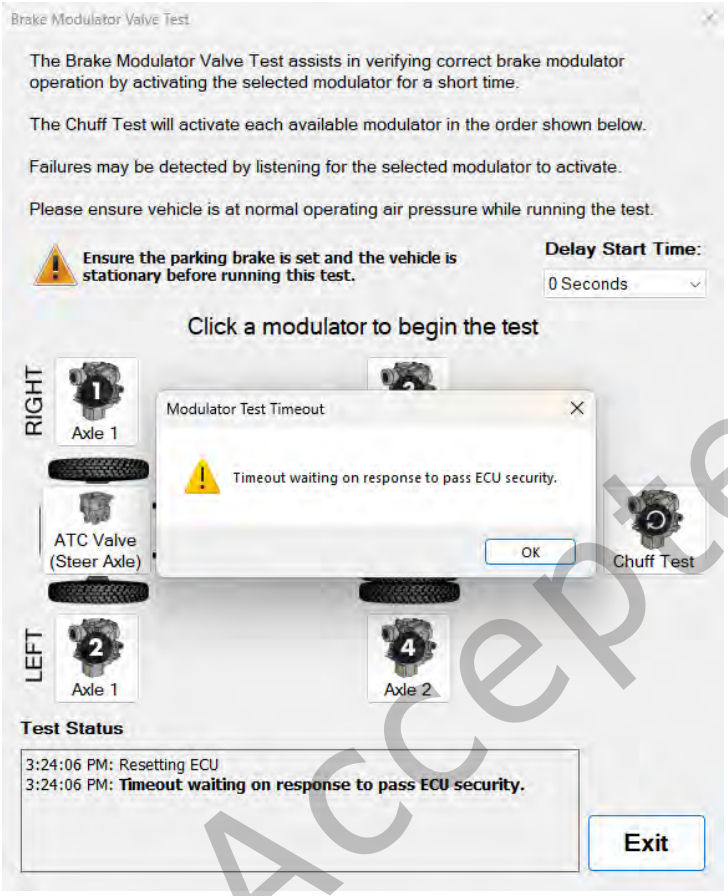

Fig. 20. Session Downgrade Impact - Bendix A-COM Diagnostic Tool

6.3 Session Downgrade Vulnerability

In this experiment, we investigate the *Session Downgrade Vulnerability*, hypothesizing that requesting a lower-level session while an ECU is actively in a programming or extended diagnostic session could cause the ECU to terminate the current session and revert to a default session. This downgrade could disrupt critical functions such as firmware updates, diagnostics, or reprogramming, potentially leading to unintended behavior.

6.3.1 Hypothesis. According to the ISO 14229-1 standard, a higher-level diagnostic session can be left by requesting a default session or hard reset. It further states that upon receiving a default session request, it shall close all functionality exposed in the current session and return to a default state. We hypothesize that if an attacker sends a request for a default diagnostic session (0x01) while the ECU is engaged in a more privileged session, such as a programming session (0x02) or an extended diagnostic session (0x03), the ECU will downgrade the active session, terminating any ongoing tasks such as reprogramming or diagnostics. This is illustrated in Fig. 19 could introduce significant security and operational risks, particularly during critical phases such as software updates.

Fig. 21. Session Downgrade Impact - Wabco Toolbox Diagnostic Tool

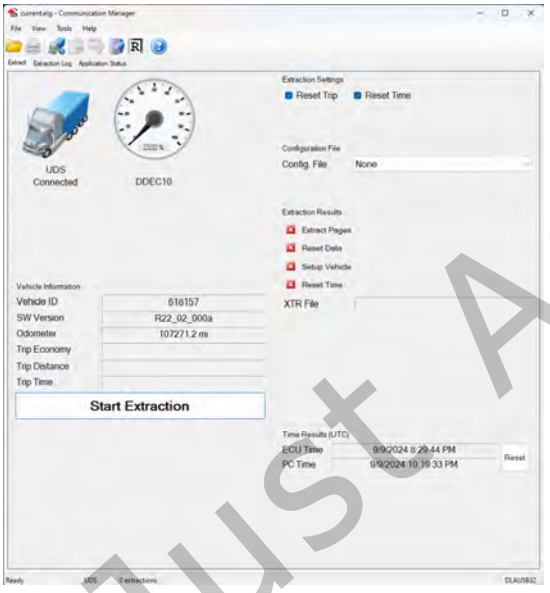

Fig. 22. Normal Session - DDEC Reports Diagnostic Tool (Testbeds 3,4)

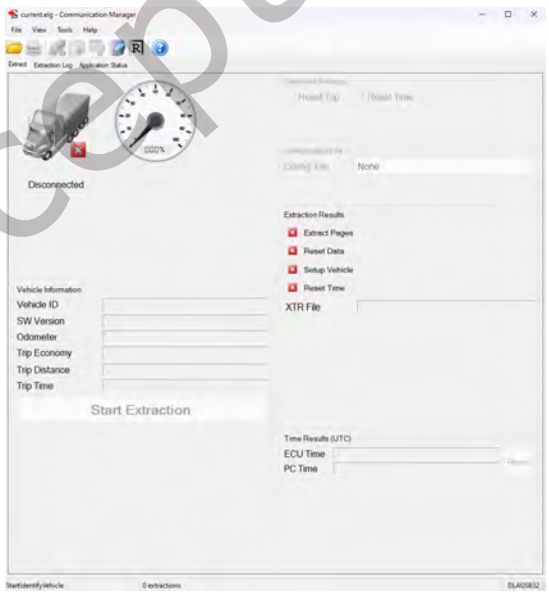

Fig. 23. Session Downgrade Impact - DDEC Reports Diagnostic Tool (Testbeds 3,4)

6.3.2 Testing. The attack was conducted on both the local testbeds and the Freightliner Cascadia testbed. An active extended diagnostic session was first initiated with the ECU using a legitimate Diagnostic Session Control message (0x10 0x03), typically used for accessing memory parameters. Then, from a separate device, a Diagnostic Session Control message requesting the default session (0x10 0x01) was sent. During this time, attempts were made to continue the extended diagnostic session from the legitimate diagnostic tool.

6.3.3 Results and Observations. Each testbed's ECU immediately terminated the active extended diagnostic session and reverted to the default session upon receiving the session downgrade request. The ongoing data exchange process was abruptly halted, and all diagnostic tools connected to the ECU lost their session. This behavior was consistent across multiple testbeds. Figures 20 and 21 display the impact on local testbeds 1 and 2, where both diagnostic tools were disrupted and unable to obtain authorization for an extended diagnostic session when the session was downgraded. Similar observations were made on testbeds 3 and 4, where before the attack the diagnostic software was connected to the ECU as shown in Fig. 22 and after the attack was launched, the tool disconnected as shown in Fig. 23 due to the session downgrade. The same was verified on the Freightliner Cascadia testbed. In all cases, the ECU reverted to the default session, preventing the original operations from being completed.

6.3.4 Possible Mitigation. Mitigating the *Session Downgrade Vulnerability* requires stricter session management controls within the ECU. One possible solution is to implement logic that prevents session downgrades while a higher-level session, such as programming or extended diagnostic sessions, is active. Additionally, session priority handling should be improved to ensure that once a programming session has been initiated, no external request can force the ECU to revert to a default session without proper authentication and confirmation. Implementing session state protection mechanisms, such as blocking session downgrade requests unless explicitly terminated by the authenticated tool, could significantly reduce the risk of this vulnerability being exploited.

6.4 Diagnostics Jam Vulnerability

Our final experiment targets the *Diagnostics Jam Vulnerability*„ in which we examine the effects of sending a rapid sequence of 'Wait' and 'Clear to Send' messages to an ECU. This approach tests whether such an attack can interfere with the ECU's normal operations, and potentially cause a service disruption.

6.4.1 Hypothesis. The ISO 15765-2 standard, commonly known as ISO-TP, enables communication and multi-packet data transfer over the transport protocol between an external diagnostics tester and a vehicle's ECU. According to the protocol specifications, Flow Control (FC) frames manage the transmission of multi-frame messages, where 'Wait' frames signal a pause in data transfer and 'CTS' frames indicate the continuation of data transmission. The protocol also mandates that an ECU must monitor the number of consecutive 'Wait' frames received in series and terminate data transfer after encountering a specified number of 'Wait' Flow Control frames in succession, as defined by the ECU manufacturer. However, this counter is reset upon receiving a CTS frame. Our hypothesis, defined in Fig . 24, suggests that sending a specific sequence of FC frames - repeatedly alternating between 'Wait' and 'CTS' frames within the maximum allowable count of 'Wait' frames - could exploit the ISO-TP flow control mechanism. This could potentially force the target ECU into a state where is becomes overwhelmed and temporarily unable to process or respond to other diagnostic requests, effectively creating a 'Diagnostics Jam' condition.

6.4.2 Testing. The attack was executed on both the local testbeds and the Freightliner Cascadia cab testbed. The testing procedure was structured as follows: A standard diagnostic tester continuously send UDS requests in a routine manner. Concurrently, our script was set to repeatedly request data for a Diagnostic Trouble Code (DTC) using the multipacket ISO-TP protocol.

The script initially waited for the FF from the ECU. Once the FF was received, an FC message with a 'CTS' signal was sent to allow data transfer. Midway through this process, we deliberately introduced a mixture of 'CTS' and 'Wait' FC frames. After reception of the FF, we transmitted an FC message with a CTS for 1 packet, followed by 10 FC frames with 'Wait' signals at 100ms intervals to avoid network flooding. This mixed sequence was maintained for a specified duration before reverting to the usual pattern of sending only CTS frames. The

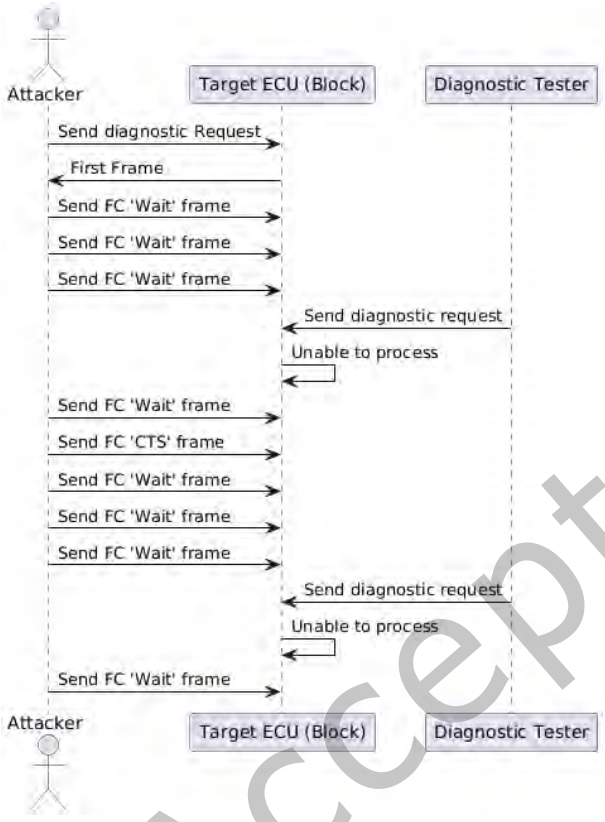

Fig. 24. Diagnostics Jam Vulnerability Hypothesis

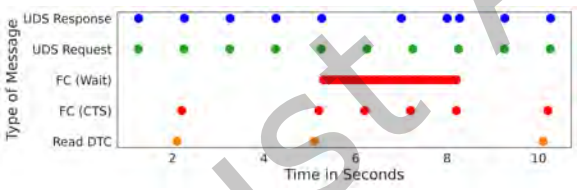

Fig. 25. Diagnostics Jam Attack Results on Local Testbed 1

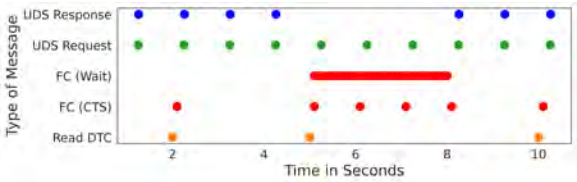

Fig. 26. Diagnostics Jam Attack Results on Local Testbed 2

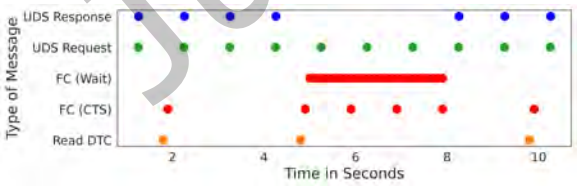

Fig. 27. Diagnostics Jam Results on Local Testbed 3

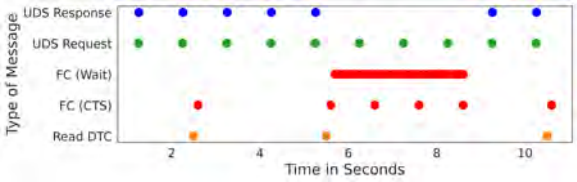

Fig. 28. Diagnostics Jam Attack Results on Local Testbed 4

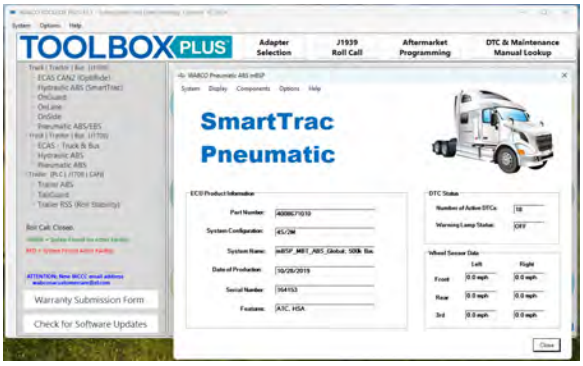

Fig. 29. Diagnostics Software during Normal Conditions

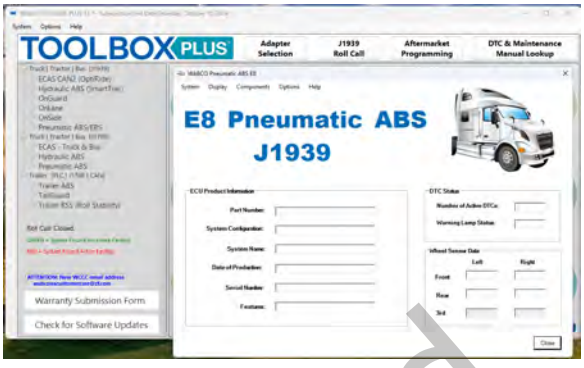

Fig. 30. Diagnostics Software during Diagnostics Jam Attack

goal was to observe how this alternating FC frame sequence would interact with and potentially disrupt the ongoing UDS requests being processed by the ECU.

6.4.3 Results and Observations. As shown in Figs. 25, 26, 27, 28, our experiments on the local testbeds revealed that the target ECU exhibited normal behavior under standard conditions, promptly responding to UDS request messages during attempts to read DTCs over multi-packet data transfer. However, when the communication involved a combination of 'CTS' and 'Wait' frames, following our proposed attack pattern, the ECU's response behavior ceased entirely.

During the attack, the target ECU entered an state of apparent overload. This was evidenced by a significant delay in responding to UDS requests on local testbed 1, and a complete failure to respond on testbeds 2-4. Essentially, the ECU entered the 'Diagnostics Jam' condition as hypothesized, where it became incapable of processing new diagnostic requests, rendering its diagnostic functionalities inoperative for the duration of the attack. Similar results were observed on the Freightliner Cascadia cab testbed. As depicted in Figs. 29, 30, under normal conditions, the diagnostic software successfully retrieved relevant information from the EBC. However, under attack, the diagnostic software failed to obtain the same information from the EBC, further validating the effectiveness of the 'Diagnostics Jam' attack.

6.4.4 Possible Mitigation. To mitigate the vulnerabilities identified within the ISO 15765-2 standard, specifically against the 'Diagnostics Jam' attack, a multifaceted approach is recommended. This approach should include adjusting protocol timeouts to counteract abnormal flow control conditions, thereby preventing the ECU from being stuck in an unresponsive state. Additionally, implementing anomaly detection algorithms to identify and respond to suspicious patterns of FC frames can help in recognizing and mitigating such attacks in real-time. Furthermore, enhancing the ECU's processing capabilities to more efficiently handle high volumes of FC frames could improve its resilience against potential overload conditions, ensuring continued operation and response to legitimate diagnostic requests.

6.5 Gateway ECU Testing in the Freightliner Cascadia

Our investigation into the Freightliner Cascadia's network systems involved a comprehensive analysis of the gateway ECU. Within our experimental setup, we transmitted a range of diagnostic and attack messages to monitor the gateway ECU's response. Specifically, messages were sent from the diagnostic CAN network, and we surveyed their transfer to the internal ECU network. This process was essential for interpreting the results

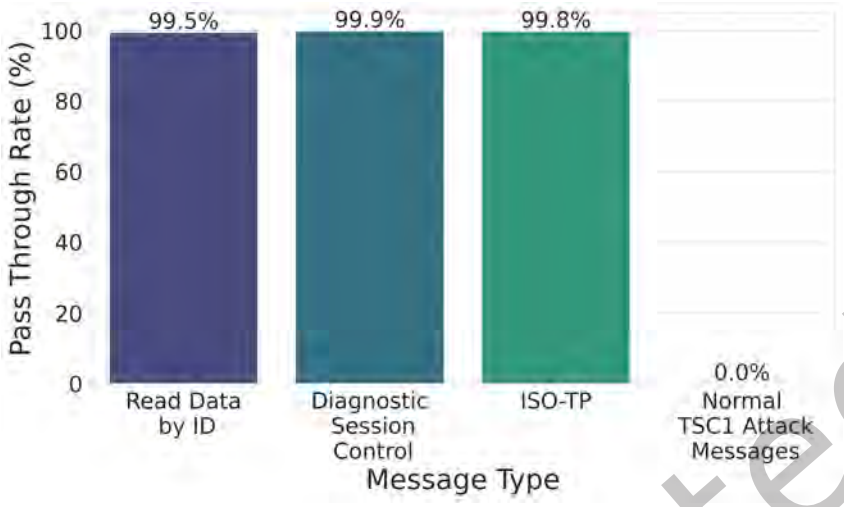

Fig. 31. Behavior of the Gateway ECU in Filtering Different Message Types

presented in Fig. 31, providing insight into how the gateway ECU manages and filters network traffic under various conditions.

The results of these tests indicate a noticeable difference in how the gateway ECU handles assorted message types. It consistently allowed nearly all diagnostic messages including those related to 'Read Data by ID', 'Diagnostic Session Control', and 'ISO-TP'. In contrast, it successfully blocked all standard J1939 Torque/Speed Command 1 (TSC1) attack messages, which were previously demonstrated by Burakova et al. [3] in their attack scenario. This suggests that while the gateway ECU has robust filtering mechanisms for certain high-risk commands, it remains permissive towards diagnostic messages, highlighting a potential area of vulnerability. This evidence is consistent with the findings discussed in the previous sections on the 'Read Data by ID Vulnerability,' 'Session Denial Vulnerability,' and 'Diagnostics Jam Vulnerability.' Each of these vulnerabilities were successfully executed on the Freightliner Cascada's system, resulting in a noticeable impact on its ECUs.

7 CONCLUSION AND FUTURE WORK

This paper presents four distinct scenarios where protocol vulnerabilities in UDS standards may be exploited to expose ECUs in Medium and Heavy-Duty Vehicles to various types of attacks. The first three scenarios reveal novel vulnerabilities with the ISO 14229 standard, while the fourth scenario introduces a new attack vector exploiting the ISO 15765 (Diagnostic Communication over CAN) specifications.

The use of protocols such as SAE J1939, ISO 15767 and ISO 14229 is not strictly mandated in MHD vehicle systems. However, adherence to these protocols has become an effective method to ensure interoperability with other ECUs that do implement them. When these protocols are adopted, their implementation is left to the discretion of each ECU manufacturer, leading to variations in behavior and responses. As such, the vulnerabilities discussed in this paper may manifest differently across platforms. Future work should therefore focus on expanding the experimental evaluation to include a broader set of ECUs, particularly those not yet tested, to further validate and generalize the findings.

Fundamentally, this paper contributes to broadening the current understanding of the vehicle security threat landscape for Medium and Heavy-Duty Vehicles. Incorporating these scenarios into security and functional testing is crucial for identifying potential logic flaws in deployed components. A significant portion of the

networking specifications remain under-examined for security vulnerabilities, indicating ample opportunities for further research. Additionally, developing effective defense mechanisms to inhibit these attacks is of keen interest.

REFERENCES

[1] 2021. Road vehicles — Unified diagnostic services (UDS) — Part 2: Session layer services. https://www.iso.org/standard/77322.html

[2] Tyler Biggs, Rik Chatterjee, and Jeremy Daily. 2025. Forging Clean Truck Check Test Reports with a DLL Hijacking Attack. In *Proceedings of the 3rd Symposium on Vehicle Security and Privacy (VehicleSec)*. USENIX Association, Seattle, WA, USA. To appear. Presented at VehicleSec 2025, co-located with USENIX Security Symposium.

[3] Yelizaveta Burakova, Bill Hass, Leif Millar, and Andre Weimerskirch. 2016. Truck Hacking: An Experimental Analysis of the SAE J1939 Standard. In *Proceedings of the 10th USENIX Conference on Offensive Technologies*. USENIX Association, Austin, TX, USA, 211–220.

[4] Matthew Timothy Campo, Subhojeet Mukherjee, and Jeremy Daily. 2021. Real-Time Network Defense of SAE J1939 Address Claim Attacks. *SAE International Journal of Commercial Vehicles* 14, 3 (Aug. 2021), 02–14–03–0026. https://doi.org/10.4271/02-14-03-0026

[5] Rik Chatterjee. 2024. *Security Shortcomings of Embedded Network Protocols in Commercial Vehicles*. Master's Thesis. Colorado State University, Fort Collins, CO, USA. https://www.proquest.com/openview/03ddd23b3a8e0878d27ba9a2093b623a/1?pq-origsite=gscholar&cbl=18750&diss=y Available from Mountain Scholar Digital Repository.

[6] Rik Chatterjee, Carson Green, and Jeremy Daily. 2024. Exploiting Diagnostic Protocol Vulnerabilities on Embedded Networks in Commercial Vehicles. In *Proceedings of the Symposium on Vehicle Security and Privacy (VehicleSec)*. USENIX Association, San Diego, CA, USA. https://www.nsf.gov/publications/pub_summ.jsp?ods_key=exploiting_diagnostic_protocols Cited by 7 as of 2025.

[7] Rik Chatterjee, Ben Karel, Ricardo Baratto, Michael Gordon, and Jeremy Daily. n.d.. Assured Micropatching of Race Conditions in Legacy Real-time Embedded Systems. *Scholar Articles* (n.d.). Available online.

[8] Rik Chatterjee, Subhojeet Mukherjee, and Jeremy Daily. 2023. Exploiting Transport Protocol Vulnerabilities in SAE J1939 Networks. In *Proceedings of the Inaugural International Symposium on Vehicle Security & Privacy*. Internet Society, San Diego, CA, USA. https://doi.org/10.14722/vehiclesec.2023.23053

[9] Rik Chatterjee, Subhojeet Mukherjee, and Jeremy Daily. n.d.. Transport Layer Vulnerabilities in the SAE J1939 Protocol-Request Overload. *Scholar Articles* (n.d.). Available online.

[10] Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, Stefan Savage, Karl Koscher, Alexei Czeskis, Franziska Roesner, and Tadayoshi Kohno. 2011. Comprehensive Experimental Analyses of Automotive Attack Surfaces. In *USENIX Security Symposium*, Vol. 4. USENIX Association, San Francisco, CA, USA, 447–462.

[11] Kyong-Tak Cho and Kang G Shin. 2016. Fingerprinting Electronic Control Units for Vehicle Intrusion Detection. In *Proceedings of the 25th USENIX Conference on Security Symposium (SEC'16)*. USENIX Association, Austin, TX, USA, 911–927.

[12] Chandrima Ghatak. 2024. Toward Robust Embedded Networks in Heavy Vehicles-Machine Learning Strategies for Fault Tolerance. Available online.

[13] Chandrima Ghatak, Rik Chatterjee, Martin Trae Span, and Jeremy Daily. 2024. A Systems Approach for Designing Open Vehicle Data Archiving Systems. In *2024 IEEE International Symposium on Systems Engineering (ISSE)*. 1–8. https://doi.org/10.1109/ISSE63315.2024.10741089

[14] Chandrima Ghatak, Saira Jabeen, Hossein Shirazi, and Indrakshi Ray. 2023. Improving the Resiliency of Embedded Networks in Heavy Vehicles-Towards Fault Tolerance. In *Proceedings of the Ninth Annual Industrial Control System Security (ICSS) Workshop. Annual Computer Security Applications Conference (ACSAC)*. ACSAC.

[15] Carson Green, Rik Chatterjee, and Jeremy Daily. 2025. Persistent Firmware-Level Compromise in a Maritime Autopilot System. In *Proceedings of the 3rd Symposium on Vehicle Security and Privacy (VehicleSec)*. USENIX Association, Seattle, WA, USA. To appear. Presented at VehicleSec 2025, co-located with USENIX Security Symposium.

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

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

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

[19] Jake Jepson, Rik Chatterjee, and Jeremy Daily. 2024. Commercial Vehicle Electronic Logging Device Security: Unmasking the Risk of Truck-to-Truck Cyber Worms. In *Symposium on Vehicles Security and Privacy (VehicleSec)* (26 February 2024). VehicleSec Symposium, San Diego, CA, USA. https://doi.org/10.14722/vehiclesec.2024.23047

[20] Jake Jepson, Rik Chatterjee, and Jeremy Daily. 2024. Demo: Exploiting Cybersecurity Flaws from the ELD Mandate for Trucks. In *Proceedings of the Symposium on Vehicle Security and Privacy (VehicleSec)*. Internet Society, San Diego, CA, USA. https://www.ndss-

symposium.org/wp-content/uploads/vehiclesec2024-9-demo.pdf
[21] Karl Koscher, Alexei Czeskis, Franziska Roesner, Shwetak Patel, Tadayoshi Kohno, Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, and Stefan Savage. 2010. Experimental Security Analysis of a Modern Automobile. In *IEEE Symposium on Security and Privacy*. IEEE, Oakland, CA, USA, 447–462.
[22] Sekar Kulandaivel. 2021. *Revisiting Remote Attack Kill-Chains on Modern In-Vehicle Networks.* PhD thesis. Carnegie Mellon University. Available at: https://kilthub.cmu.edu/ndownloader/files/34106900.
[23] Sharika Kumar, Jeremy Daily, Qadeer Ahmed, and Anish Arora. 2023. Cybersecurity Vulnerabilities for Off-Board Commercial Vehicle Diagnostics. In *Proceedings of the WCX SAE World Congress Experience*. 12. https://doi.org/10.4271/2023-01-0040
[24] John Maag, Christopher Reding, and Kelly Howell. 2017. Seed-Key Security Exchange. Presented at the 2017 Heavy Vehicle Cyber Security Workshop sponsored by the National Motor Freight Traffic Association, Inc..
[25] Charlie Miller and Chris Valasek. 2014. A Survey of Remote Automotive Attack Surfaces. In *Black hat USA*. Blackhat Press, Las Vegas, NV, USA, 94.
[26] Charlie Miller and Chris Valasek. 2015. Remote Exploitation of an Unaltered Passenger Vehicle. In *Blackhat USA*. Blackhat Press, Las Vegas, NV, USA.
[27] Subhojeet Mukherjee, Rik Chatterjee, and Jeremy Daily. 2025. TruckSentry: Context Aware Intrusion Detection and Prevention System for J1939 Networks. *IEEE Open Journal of Intelligent Transportation Systems* 6 (2025), 294–309. https://doi.org/10.1109/OJITS.2025.3545474
[28] S. Mukherjee, H. Shirazi, I. Ray, J. Daily, and R. Gamble. 2016. Practical DoS Attacks on Embedded Networks in Commercial Vehicles. In *Proceedings of the 12th International Conference on Information Systems Security* (Jaipur, India). 23–42.
[29] P. Murvay and B. Groza. 2018. Security Shortcomings and Countermeasures for the SAE J1939 Commercial Vehicle Bus Protocol. *IEEE Transactions on Vehicular Technology* 67, 5 (2018), 4325–4339.
[30] Teddy Nyambe, Rik Chatterjee, and Jeremy Daily. 2025. Short Paper: Software Bill of Materials Management for Embedded Vehicle Systems. In *Security and Privacy in Cyber-Physical Systems and Smart Vehicles. SmartSP 2024 (Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, Vol. 622)*, Xiali Hei, Luis Garcia, Taesoo Kim, and KyoungSoo Kim (Eds.). Springer, Cham, 100–109. https://doi.org/10.1007/978-3-031-93354-7_5
[31] Habeeb Olufowobi, Sena Hounsinou, and Gedare Bloom. 2019. Controller Area Network Intrusion Prevention System Leveraging Fault Recovery. In *Proceedings of the ACM Workshop on Cyber-Physical Systems Security & Privacy - CPS-SPC'19*. ACM Press, London, United Kingdom, 63–73.
[32] Robert Bosch GmbH. 1991. *CAN Specification.* Standard 2.0. Robert Bosch GmbH.
[33] Society of Automotive Engineers. [n. d.]. SAE J1939 Standards Collection. https://www.sae.org/standardsdev/groundvehicle/j1939a.htm Accessed: 2021-11-09.
[34] SystemsCyber. 2023. Network Segmentation Analysis. https://github.com/SystemsCyber/NetworkSegmentationAnalysis Accessed: 2024-09-14.
[35] Marko Wolf, André Weimerskirch, and Christof Paar. 2004. Security in Automotive Bus Systems. In *Proceedings of the Workshop on Embedded Security in Cars*. Springer-Verlag, Bochum, Germany, 1–13.

Received 15 September 2024; revised 8 May 2025; accepted 4 August 2025