UTHP: The Ultimate Truck Hacking Platform

Abstract

The cybersecurity landscape for Medium- and Heavy-Duty (MHD) vehicles presents unique challenges due to complex proprietary protocols, minimal existing security measures, and a lack of dedicated, integrated tools for vulnerability testing. These vehicles, which rely heavily on protocols like J1939, J1708, and J2497, are increasingly targeted by attacks that exploit weak access controls and lack of encryption, posing serious risks to vehicle safety and functionality. The Ultimate Truck Hacking Platform (UTHP) was developed to address these gaps, offering a unified, purpose-built solution for cybersecurity research and testing in MHD vehicles, including trucks, buses, vocational, off-road, and agricultural equipment. Based on a BeagleBone architecture with custom-printed circuit boards (TruckDuck ER), the UTHP combines a tailored Yocto image with comprehensive protocol support and manipulation tools. Key contributions of UTHP include remote diagnostic capabilities, USB gadget support, and specialized tools (e.g., TruckDevil, CanCat) that enable real-time protocol analysis, penetration testing, and fuzzing. By providing an accessible, extensible platform for both in-vehicle and offsite testing, the UTHP represents a significant advancement in addressing the cybersecurity needs of embedded systems within MHD vehicles, offering researchers and practitioners an essential tool for enhancing vehicle resilience.

Keywords: Commercial Vehicle Networks, Vehicle Cybersecurity, Yocto, Linux, Controller Area Network, Local Interconnect Network, Power Line Communications

1 Introduction

The increasing connectivity and electronic control systems in MHD vehicles have introduced a range of cybersecurity risks. While protocols like SAE J1939 [17], J1708 [16], and J2497 [18] enable essential vehicle functions, they also present attack vectors that can compromise vehicle integrity, safety, and functionality. As commercial vehicles become more dependent on these protocols, they remain susceptible to attacks that exploit weak access control, lack of encryption, and minimal security at the protocol level. Although there is a growing need for cybersecurity in this sector, there are limited unified tools specifically designed to test and analyze the security of truck protocols and their implementation within the electronic control units (ECUs) .

The Ultimate Truck Hacking Platform (UTHP) is designed to address this gap by providing a modular, extensible solution for in-depth protocol testing, penetration testing, and vulnerability analysis. Built on a BeagleBone architecture with custom PCBs (TruckDuck ER), UTHP combines a tailored Yocto image with essential protocol-specific tools. By supporting protocols crucial to truck operations, including J1939, CAN [19], J1708, J1587 [15], J2497, and LIN [8], the UTHP offers unique capabilities for diagnosing, testing, and hardening the cybersecurity of MHD vehicles for applications such as freight, transport, agriculture, and emergency services.

This paper presents UTHP and its contributions to truck cybersecurity as follows:

1. **Integrated Cybersecurity Platform for Trucks**: UTHP is the first unified platform combining both hardware and software specifically tailored for truck hacking, enabling researchers and practitioners to conduct comprehensive security testing on MHD vehicles.
2. **Protocol Support and Manipulation Tools**: UTHP includes a set of protocol-specific tools (e.g., TruckDevil, CanCat) that facilitate the manipulation and analysis of key protocols such as J1939, J1708, and J2497. These tools allow for precise control over protocol messages and behaviors, which is essential for identifying security vulnerabilities in truck ECUs.

3. **Remote Access and Diagnostic Capabilities**: UTHP provides innovative remote diagnostic features, including USB gadget support and TCP/UDP server configurations, enabling real-time access and control over protocol interactions. This capability allows for effective offsite security testing, an important feature for large-scale deployment scenarios.
4. **Modular and Extensible Design**: The platform is designed to be extensible, supporting additional protocols and features as required. This modularity makes UTHP a versatile tool, adaptable to evolving cybersecurity needs within the MHD vehicle sector. It's also built from the ground up with a focus on backward compatibility with it's predecessor boards [20].

By addressing the specific security needs of MHD vehicles, UTHP presents itself as a critical tool for advancing the cybersecurity posture of embedded systems within the commercial vehicle industry.

Figure 1: UTHP Hardware Enclosure and PCB

2 Background and Related Work

The cybersecurity landscape of commercial vehicles is marked by vulnerabilities across various layers of the SAE J1939 protocol. Research by Burakova et al. [3] identified significant security weaknesses within the application layer, showing that an attacker could exploit specific J1939 messages to maintain continuous control over critical vehicle functions. Their study demonstrated that vital components, such as engine braking and accelerator input, could be compromised, posing substantial risks to the operational safety of commercial vehicles.

Focusing on the data-link layer, Mukherjee et al. and Chatterjee et al. [11, 6, 5, 4] highlighted vulnerabilities where excessive request messages directed at an Electronic Control Unit (ECU) could overload its processing capacity. Their findings also noted that unauthorized, persistent connections to an ECU could deny legitimate requests, effectively creating a denial-of-service scenario.

In the domain of network management, Murvay et al. [12] demonstrated how network integrity could be compromised through address claim message flooding, which could incapacitate multiple ECUs. They further showed that by abruptly terminating multi-packet data transfers, an attacker could disrupt ECU operations, leading to a denial-of-service condition.

Legacy protocols, such as the J2497 protocol, widely used in trailer air brake systems, has been discovered to be vulnerable to both wired and wireless read and write access [2]. This research shows that its signals can be remotely read and induced using low-cost equipment, enabling unauthorized diagnostic and control commands without authentication, and discusses possible mitigations and lessons for next-generation tractor-trailer interfaces.

Overall, various SAE Protocols, foundational to the cyber-physical systems in commercial vehicles, has been shown to contain security gaps across its layered structure [3, 11, 12, 6, 9]. Studies targeting various levels of the protocol, from application to network management and data-link layers, reveal critical security

flaws, emphasizing the need for tools like the UTHP to address these cybersecurity challenges in a unified and systematic manner.

3 Platform Overview

3.1 Hardware Design

The Ultimate Truck Hacking Platform (UTHP) has been designed to provide comprehensive interfacing capabilities with MHD truck vehicle networks, serving both diagnostic and cybersecurity functions. Central to its design is the facilitation of connectivity across multiple communication protocols prevalent in MHD vehicles, including Controller Area Network (CAN), Local Interconnect Network (LIN), J1708, and J2497 standards. The hardware enclosure can be seen in Figure 1.

The UTHP is based on the BeagleBone Black (BBB) [1] single-board computer, selected for its processing capabilities and numerous General Purpose Input/Output (GPIO) pins, used for interfacing with the software suite and hardware present onboard. Each function within the UTHP directly addresses stakeholder needs, such as enabling communication with multiple vehicle networks, supporting specific software packages, and meeting interface requirements—including connectors and compatibility with various vehicle network protocols. To fulfill these requirements, the platform incorporates a suite of advanced features:

1. **Bitmagic Basic Logic Analyzer**: This cost-effective tool provides live signal monitoring, capturing and analyzing signal transitions and timing within vehicle networks. It is essential for in-depth diagnostics and cybersecurity analyses.
2. **Safe Shutdown Mechanism**: To prevent data corruption in the event of power disruptions—such as accidental cable removal—the UTHP includes a safe shutdown feature by utilizing a super-capacitor. This mechanism ensures the operating system and data remain intact during unexpected power loss.
3. **Real-Time Clock (RTC)**: The RTC supports time-sensitive operations, enhancing the accuracy of data logging crucial for precise diagnostics and historical analysis.
4. **Provisions for a Hardware Security Module**: the design accommodates an ATECC608A-SSHDA-B hardware security module to enable future security enhancements, such as secure boot processes and cryptographic key management.
5. **Mikroe Click Compatibility**: Offers modular expandability through access to a vast array of available modules. This feature future-proofs the system, allowing it to adapt to evolving technological demands without requiring significant hardware modifications.
6. **Interfaces**: The UTHP provides options compatible with Deutsch 9-pin diagnostic connectors and standard laptops. These connections enable real-time data logging, analysis, and system configuration, ensuring seamless integration into existing diagnostic setups and supporting a wide range of vehicle communication protocols.
7. **Power Management**: The UTHP accommodates both USB power and direct 12V vehicle power sources. A reliable voltage regulation system ensures a stable 5V output from a 12V input, incorporating decoupling capacitors and Transient Voltage Suppression (TVS) diodes for enhanced power stability and protection against voltage spikes. Buffers and inverters are strategically employed to protect the BBB's input/output pins during boot sequences and to manage voltage level conversions between 5V and 3.3V logic levels, enhancing the overall robustness of the hardware design.
8. **Debugging Capabilities**: To support development, debugging, and testing processes, the UTHP includes multiple UART/serial debug headers, through-hole test points, and diagnostic LEDs that provide real-time visual feedback on system status and operations.

The UTHP's communication interfaces are extensively designed to support a multitude of protocols: CAN, LIN, J1708, J1587, J2497, UART, Ethernet, TCP/IP, SPI, and I^2C—thereby providing significant flexibility in connecting to various truck systems and peripherals. The arrangement of these interfaces allows for redundant and versatile connectivity options, facilitating simultaneous interactions with the truck, a laptop, and the UTHP itself. This is further augmented by the inclusion of multiple USB ports and an Ethernet port, with panel mounts for internal reliability, and banana jack breakout boards providing access to key signals such as LIN, J1708, multiple CAN channels, and battery +12V and GND.

In terms of physical design, the UTHP is enclosed within a ruggedized Hammond 1455QPLBK-10 aluminum enclosure measuring 220 mm in length, 125 mm in width, and 51.51 mm in height. This sturdy

casing not only protects the internal components from the typical environmental stresses encountered in vehicular applications, such as vibration and physical impact but also provides valuable information to the user visually through laser-engraved documentation in regard to pinouts and interface uses. The design allows for the printed circuit board (PCB) to securely slide into the enclosure, with end panels locking it in place, ensuring structural integrity during operation.

The UTHP represents a comprehensive, flexible, and forward-thinking hardware solution for interfacing with MHD truck networks. It integrates robust hardware components, provides multiple communication interfaces, thoughtful power management, and provisions for future security enhancements, all encapsulated within a durable physical enclosure suitable for challenging vehicular environments. This platform not only meets current diagnostic and cybersecurity demands but is also poised to adapt to the evolving technological landscape of vehicular systems.

3.2 Software Stack

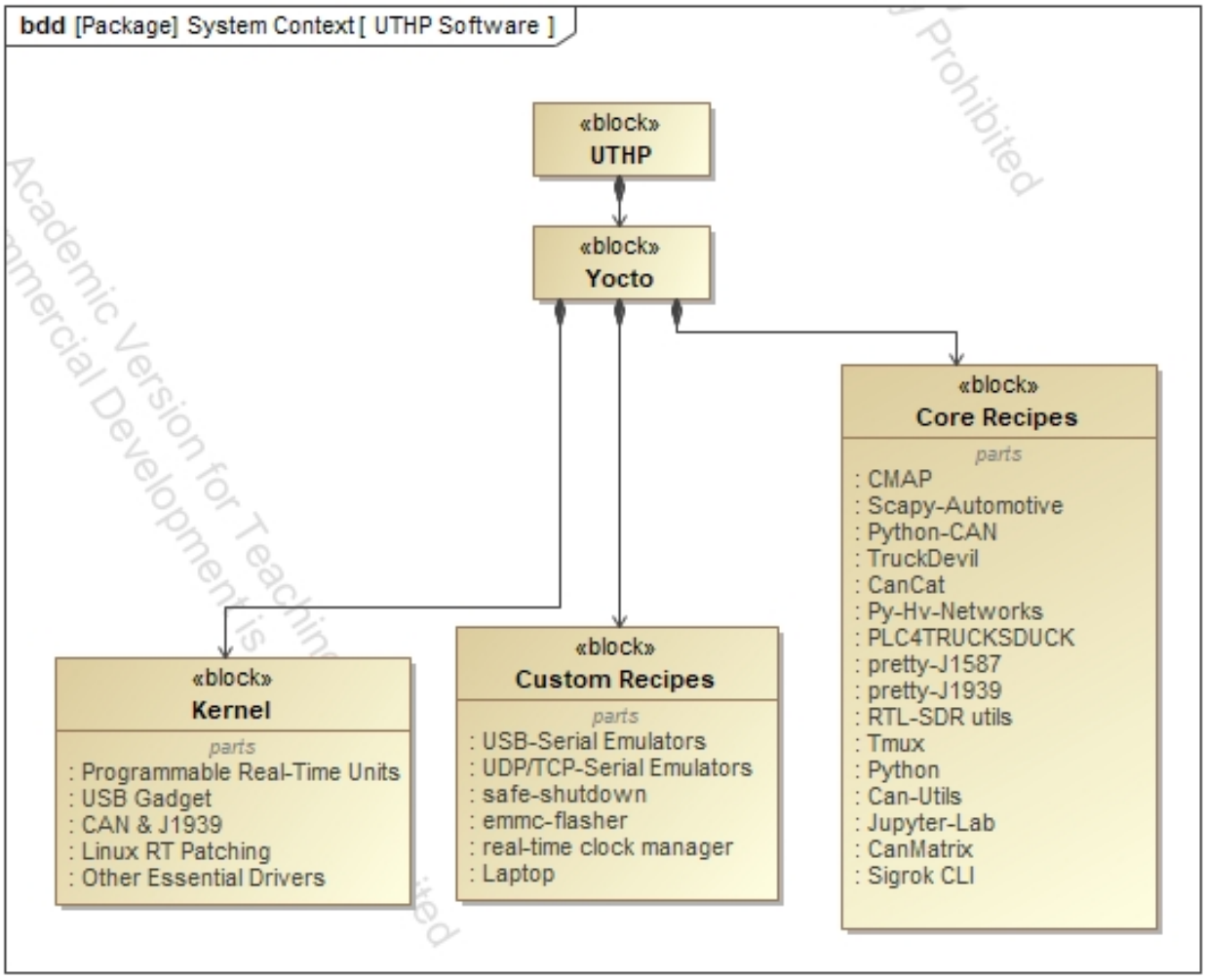

Figure 2: UTHP Software Stack SysML Context Diagram

At the core of the UTHP's software stack is the Yocto Project. The Yocto project is an embedded Linux framework which was found to work in an automotive security context for its high customization and its feature-rich set of software packages [13]. Software in the automotive industry is growing quickly, [14] so incorporating the Yocto Project's long-term support, combined with regular kernel updates and patches creates a long-term, unified, CAN diagnostics system, and cybersecurity research tool. The Yocto project

provides software in the form of recipes, so in the context of this paper, 'recipes' are provided in the form of apt packages during the final build of the operating system.

Figure 2, demonstrates the tooling available for users, as well as a general overview of the operating system. It's important in the realm of cyber-physical systems to model the systems and components, because not only does it help manage the complexity of the UTHP, but also showcase the software present. This can be used to quickly model attack vectors, and prevent cybersecurity attacks. The UTHP includes custom software developed for the Programmable Real-Time (PRU) units of the BBB. This includes custom device tree overlays used to facilitate communicate between remote processors like the AM3558 Sitara and the PRU. The PRUs are well suited for real-time communication and data collection, especially useful in MHD automotive applications. The 200MHz dedicated processing speed of the PRUs are more than enough to precisely monitor inter-byte gaps of communication protocols such as SAE J1708. The UTHP kernel is configured with support for J1939 and CAN, enabling high-performance kernel-level processing within the operating system.

The operating system is built such that it can fit onto any 4GB micro sd card, or with custom flashing software onto the 4GB Embedded MultiMediaCard (eMMC) found within the UTHP. With Yocto, there is a focus on legacy system interoperability and future proofing. This means that the operating system built for the UTHP today, is supported on other BeagleBone models and legacy capes.

As described previously, the UTHP includes a super-capacitor. Implementing a custom safe-shutdown program that monitors the Power Management Integrated Circuit (PMIC) on the BBB enhances the UTHP's data retention capabilities, ensuring reliability in the event of an operating system or operator failure.

Sometimes, the UTHP operator may not have the technical skills or knowledge necessary to utilize the full command line interface of the UTHP. To simplify the process of reading and writing data, the UTHP forwards encoded data to external tools such as TruckDevil [10], without the need to interact with the command line interface of the UTHP.

Lastly, the UTHP is comprised of a fine-grained set of core-recipes, or otherwise tools:

1. **Python3 and 2.7:** Both Python 3.12 and Python 2.7 (for legacy system interoperability) serve as the dominant programming language in quick, and rapid prototype / development.
2. **CMAP:** A Python library designed for reverse engineering and analyzing proprietary CAN-based vehicle ECUs. It facilitates the identification of undocumented diagnostic commands, and services.
3. **Scapy Automotive:** Python-based analysis of packets for protocols like J1939, CAN, and UDS. Used commonly for simulating ECUs or diagnostic tools to test vehicle behavior during penetration testing or protocol fuzzing.
4. **Python-CAN:** Python-based library for development of full-duplex CAN messaging. Commonly used for it's abstraction, and automation in logging, injecting, and simulating CAN messages.
5. **TruckDevil:** A Python, module-based framework assessing ECUs that use J1939 for communications.
6. **CanCat:** Serves as a multi-purpose CAN research tool. It supports capturing and transmitting messages on a CAN bus analyzing and identifying messages, and sharing data across different vehicle manufacturers.
7. **CAN-utils:** CAN bus network utilities, such as 'candump', 'cansniffer', and 'canbusload'. It is part of the socketcan kernel interface, allowing for real-time CAN packet capture, data collection, and analysis.
8. **Py-HV-Networks:** Python interfaces to TruckDuck interfaces, used in analyzing SAE J1708 and J1587 protocols. This includes utils such as j1708dump, similar to 'candump'.
9. **PLC4TrucksDuck:** PRU firmware used in reverse engineering PLC (SAE J2497) packets commonly found in heavy-duty trailers and Electric Vehicle (EV) charging.
10. **Pretty J1939:** Prints J1939 messages in a pretty format, useful in decoding.
11. **Pretty J1587:** Prints J1587, J1708, and J2497 messages in a pretty format, useful in decoding.
12. **RTL-SDR Utils:** Command line utility for connecting to a Software-Defined Radio (SDR). Commonly used for capturing wireless signals such as the Tire Pressure Monitoring System (TPMS), and analyzing data for interface issues or security vulnerabilities.
13. **TMUX:** Terminal Multiplexing, enables the management of multiple terminal sessions, beneficial for running data collection and analysis tasks in parallel.
14. **Jupyter Lab:** Dominant interactive environment for Python data analysis, promoting collaborative research and development.
15. **CANMatrix:** A Python-based CAN Matrix Object, used in describing ECU signals and values. Also

includes command-line tools for converting and comparing CAN databases, aiding in the adaption of OEM CAN configurations.

16. **Sigrok CLI:** Used for capturing and decoding signal waveforms from various communication interfaces. Useful in decoding low-level electrical issues, or for research before bit banging an automotive network.

4 Implementation

The UTHP OS, known as TruckHacking OS, is based on the Yocto-Poky Scarthgap Long-Term Support (LTS) reference distribution [22], with its providing layers:

1. **Meta-uthp:** Provides custom PRU firmware, uboot patches, kernel patches, connectivity modules, and custom tools.
2. **Meta-poky:** Provides a starting point for custom distributions and helps set up the build environment.
3. **Meta-openembedded:** Provides core packages related to typical embedded operating environments.
4. **Meta-python2:** Provides recipes for Python 2 modules and related tools.
5. **Meta-python:** Provides recipes for packaging Python modules.
6. **Meta-jupyter:** Provides recipes related to the Jupyter ecosystem, including Jupyter Lab and its dependencies.
7. **Meta-networking:** Provides a range of network-related software recipes.
8. **Meta-arm:** Provides a general layer for ARM architecture support.
9. **Meta-ti:** Provides Board Support Package (BSP) recipes and configurations for Texas Instruments (TI) hardware platforms.

These layers collectively ensure the required functionality, efficiency, and performance metrics are met for the UTHP [21].

Algorithm 1: PRU Firmware Pseudo Code

Input: Transmit buffer (`transmitBuf`), Receive buffer (`receiveBuf`)
Output: Messages transmitted or received via PRU RPMsg

```
Initialize PRU RPMsg Transport and Buffers;
while True do
    if Transmit buffer not empty or new message received from host then
        if Bus is idle then
            Transmit first byte (MID);
            if Another message is received and our MID loses arbitration then
                Read remaining message from bus;
                Send received message to host;
            else
                Transmit remaining bytes of the message;
                Clear transmit buffer;
    else if Message detected on the bus then
        Read the complete message;
        Send the received message to the host;
        Clear the receive buffer;
```

With the UTHP, standards for decoding MHD vehicles are built in. Many open-source tools make use of these standards to decode SAE J1939, J1708, J1587, and J2497. An example of the UTHP decoding J1708 messages can be seen in Figures 3 and 5, where the UTHP takes in raw J1708 messages, and each Message Identification (MID) and Parameter Identification (PID) is decoded into a human-readable format. This allows technicians to decode fault codes without the use of proprietary softwares. Figure 5 further demonstrates the control flow of decoding J1708 messages. The PRU runs custom firmware shown in algorithm 1 that parses bytes until a full J2497 or J1708 message is compiled. Each message is passed from the real-time decoder built into the PRU, through kernel level drivers that pass raw packets facilitated by

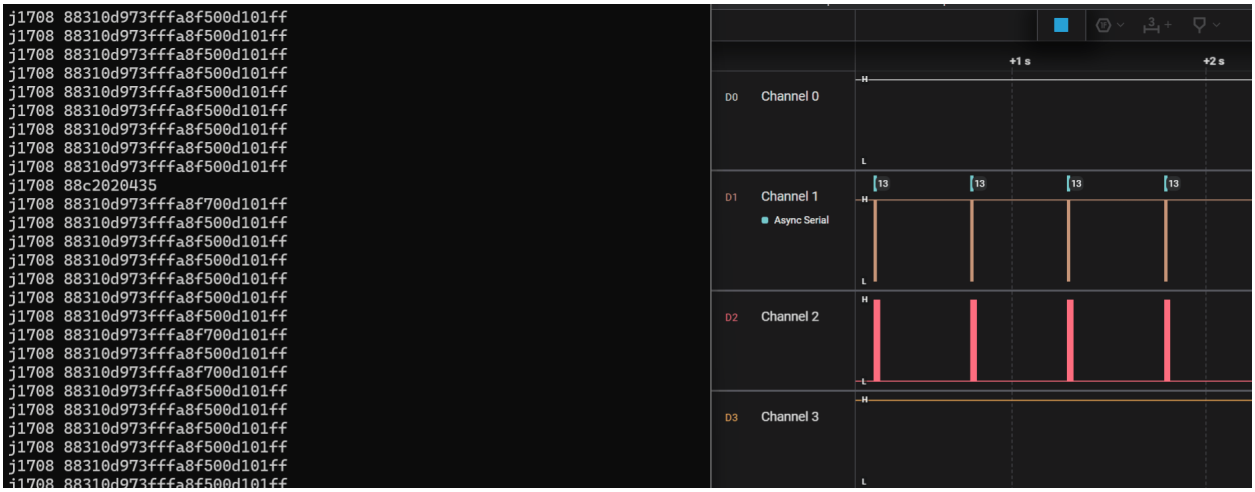

Figure 3: Wabco Brake Controller J1708 Messages Verified by j1708dump (left) and an Oscilloscope (right)

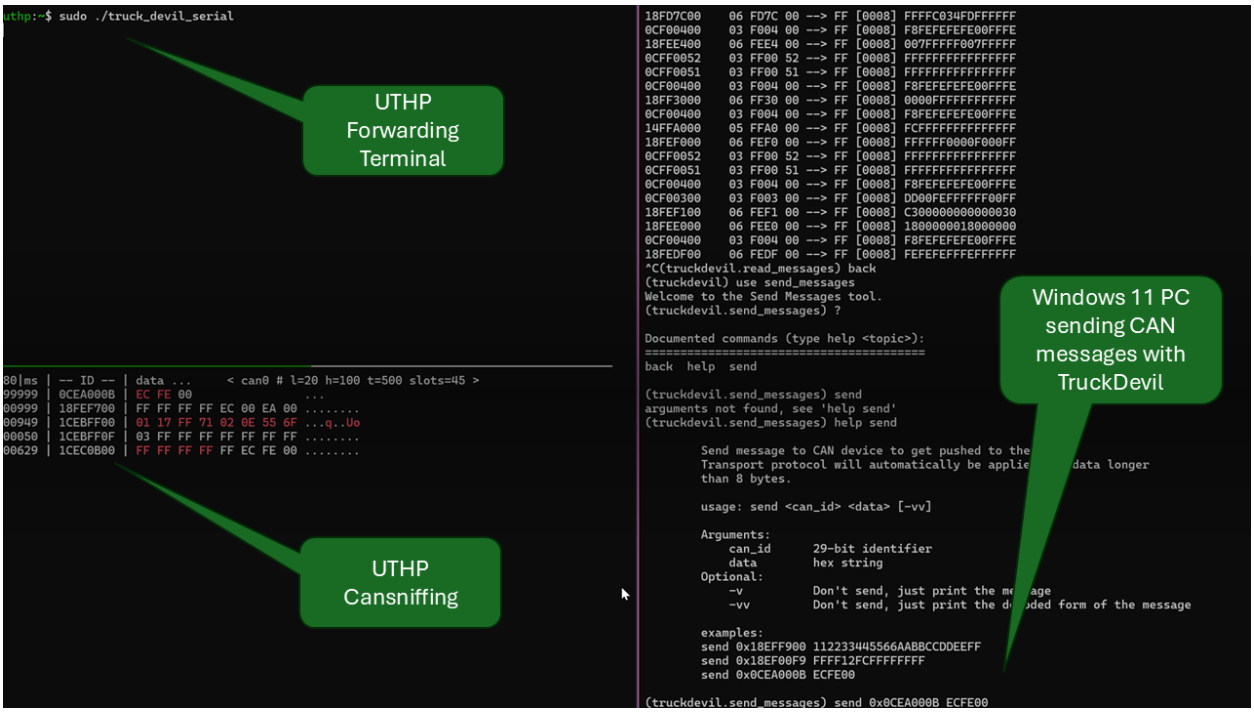

Figure 4: UTHP and Windows TCP/serial remote diagnostics output

a mailbox inter-process communication system. The file descriptor in the user space allows for developers to communicate with tools such as the j17084truckduck_host user level communication system, which then broadcasts packets on a port. Tools such as Py-Hv-Networks and Pretty-J1587 have sockets that bind to that port. Further demonstrated in Figure 5, a technician can remotely decode a j1708 fault code.

Remote diagnostics can also be performed with tools like TruckDevil on a laptop, which can decode J1939 messages into a human-readable format. This information can then be used to send out J1939 messages remotely. Figure 4 showcases the ability for an end user to remotely inject CAN messages like an address claim. The UTHP has been proven to succeed in address claim attacks, flooding the bus, and preventing Engine Control Modules (ECM) from communicating with the telemetry on the instrument cluster of a truck [21].

From a penetration testing standpoint, the workflow is expertly crafted in such a way that the whole

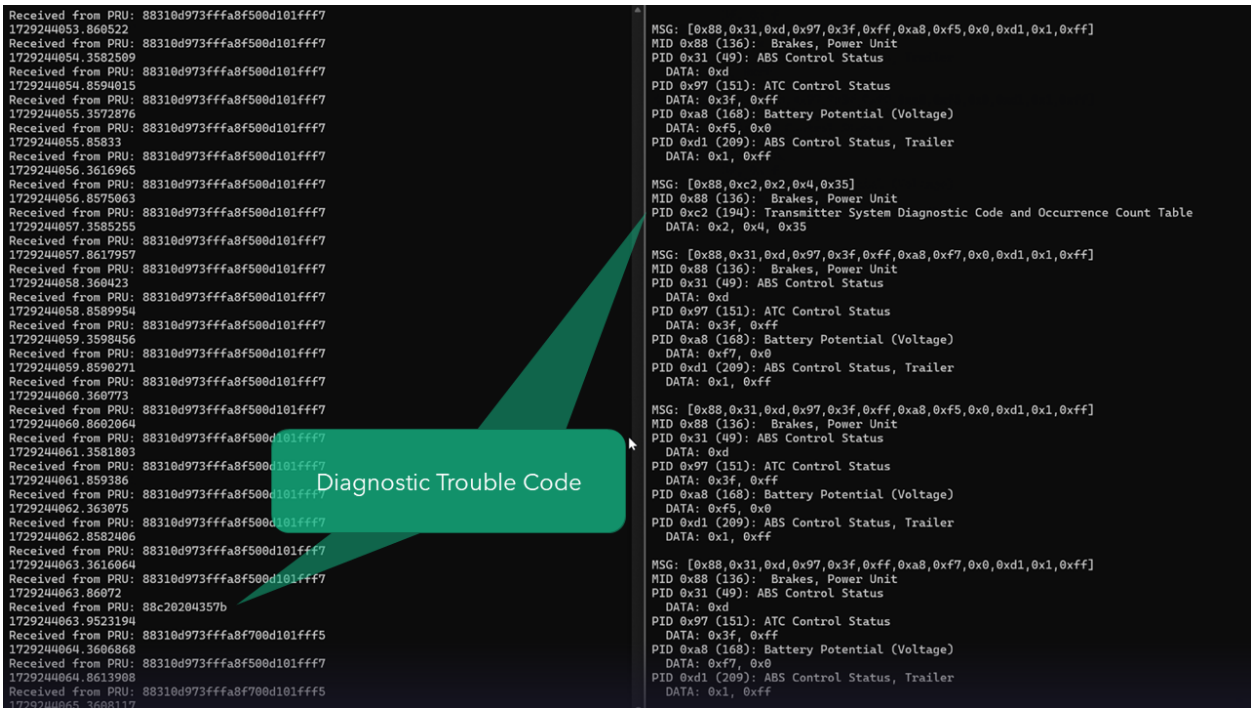

Figure 5: Translation of j1708dump output into pretty-j1587 Decoded Messages

process can be carried out on the UTHP. Tools like can-utils and CMAP allow for users to perform network enumerations, and Unified Diagnostic Services (UDS) scans, essentially performing reconnaissance on the CAN bus. Security professionals can then utilize tools like TruckDevil to fuzz J1939 messages on a bus, and finally develop exploits using python-can and jupyter notebooks to quickly report vulnerabilities.

Finally, the UTHP can serve other major functions. Given it's ability to connect to 4 can channels, it can be used to monitor the authenticity of messages between an ECM modified for Cryptographic Message Authentication Code (CMAC) calculations as shown in Figure 6, between all CAN channels present on a MHD vehicle.

5 Evaluation

5.1 Methodology

The evaluation of the UTHP's performance was conducted in alignment with the operational constraints of the BBB. Considering the single-core architecture of the BBB, testing the UTHP under multi-process workloads would not provide an equal assessment of its capabilities. Furthermore, all performance tests were executed sequentially, utilizing a single process at a time to ensure a fair and accurate evaluation of the UTHP's functionality.

5.2 Results

The initial test focused on the MCP2562 CAN bus transceiver in conjunction with the BBB Dual CAN integrated circuit controllers, under a 100% bus load condition. This evaluation employed an incremental CAN arbitration ID test to systematically validate data transmission and reception. The UTHP successfully captured every packet transmitted at full bus load from an external controller on the CAN network. This result demonstrates the UTHP's robust performance under high-load conditions, further establishing its viability as a platform for conducting security research in the MHD vehicle domain [21].

Functional validation of the J1708 bus was conducted using a Saleae Logic Pro 2 digital logic analyzer and the j1708dump utility. The Saleae Logic Pro 2, a reliable tool for analyzing diverse network signals, was employed as a reference standard to provide redundant verification of the data captured by the UTHP

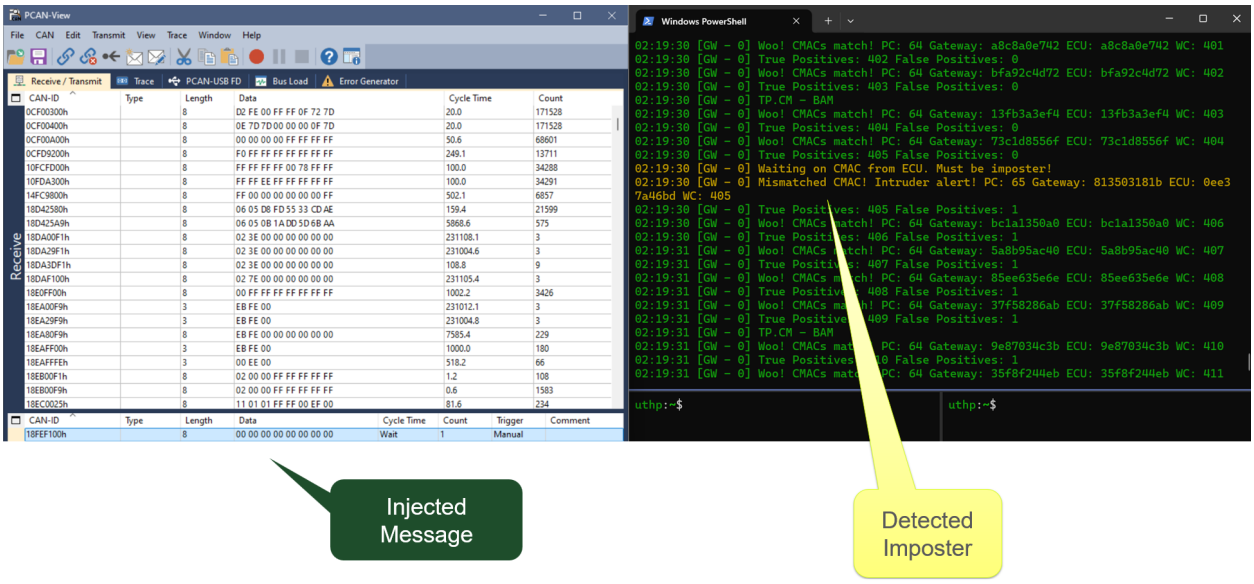

Figure 6: UTHP being utilized as a CMAC Gateway

software utilities. Comparative analysis of logs generated by the UTHP and the Saleae Logic Pro 2 revealed no discrepancies, demonstrating consistency in data capture and interpretation. Additionally, the UTHP successfully transmitted a test message, which was accurately detected and logged by the Saleae Logic Pro 2. These tests confirm the UTHP's compatibility and functionality with legacy protocols utilized in MHD (Medium and Heavy-Duty) vehicles [21].

The UTHP build was also intended to be lightweight. This means that in an evaluation of the boot times between TruckHackingOS, and the Debian operating system provided out of the box, the Yocto-based system booted 2 times faster than Debian OS [21].

6 Conclusion

The Ultimate Truck Hacking Platform (UTHP) demonstrates a significant advancement in the cybersecurity landscape for medium- and heavy-duty (MHD) vehicles. Its hardware and software integration addresses the challenges posed by legacy and proprietary vehicle protocols, offering an essential tool for diagnostics, penetration testing, and system hardening.

Future enhancements for UTHP include incorporating slcan forwarding for seamless integration with tools like SavvyCAN and implementing RP1210 standards to support broader diagnostic frameworks. Additionally, a web-based HTML landing page could enhance user interaction, providing an intuitive interface for monitoring and analysis. Security features within the Yocto Project could also be further utilized, with emphasis on secure boot processes, kernel hardening, and cryptographic module integration to elevate the platform's resilience.

To expand its real-time capabilities, future revisions will explore a PREEMPT-RT kernel configuration, aiming to improve latency and responsiveness for time-critical operations. Such updates could position UTHP as a leading solution in MHD vehicle cybersecurity, fostering innovations in vehicle diagnostics and protection mechanisms.

By continuing to evolve, the UTHP can address the growing complexity of vehicular systems, providing researchers, security professionals, and technicians with an valuable tool to secure the future of commercial vehicle networks.

7 Acknowledgments

This material is based upon work supported by the National Motor Freight and Traffic Association (NMFTA). 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 NMFTA.

References

[1] Beaglebone black single-board computer. `https://www.beagleboard.org/boards/beaglebone-black`. Accessed: 2024-08-19.

[2] Ben Gardiner. Remote writing trailer air brakes with rf. Technical report, National Motor Freight Traffic Association, IOActive, 2022. Accessed: 2024-12-16.

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

[4] Rik Chatterjee, Carson Green, and Jeremy Daily. Exploiting diagnostic protocol vulnerabilities on embedded networks in commercial vehicles. In *Symposium on Vehicles Security and Privacy (VehicleSec)*, San Diego, CA, USA, 2024. VehicleSec Symposium.

[5] Rik Chatterjee, Ben Karel, Ricardo Baratto, Michael Gordon, and Jeremy Daily. Assured micropatching of race conditions in legacy real-time embedded systems. *Scholar Articles*, n.d. Available online.

[6] Rik Chatterjee, Subhojeet Mukherjee, and Jeremy Daily. Exploiting transport protocol vulnerabilities in SAE J1939 networks. In *Proceedings of the Inaugural International Symposium on Vehicle Security & Privacy*, San Diego, CA, USA, 2023. Internet Society.

[7] Chandrima Ghatak, Rik Chatterjee, Martin Trae Span, and Jeremy Daily. A systems approach for designing open vehicle data archiving systems. In *2024 IEEE International Symposium on Systems Engineering (ISSE)*, pages 1–8, 2024.

[8] International Organization for Standardization . Road vehicles — Local Interconnect Network (LIN). Standard ISO 17987, 2016.

[9] Jake Jepson, Rik Chatterjee, and Jeremy Daily. Commercial vehicle electronic logging device security: Unmasking the risk of truck-to-truck cyber worms. In *Symposium on Vehicles Security and Privacy (VehicleSec)*, San Diego, CA, USA, 2024. VehicleSec Symposium.

[10] LittleBlondeDevil. Truckdevil, 2020. Accessed: 2024-12-16.

[11] S. Mukherjee, H. Shirazi, I. Ray, J. Daily, and R. Gamble. Practical DoS attacks on embedded networks in commercial vehicles. In *Proceedings of the 12th International Conference on Information Systems Security*, pages 23–42, 2016.

[12] P. Murvay and B. Groza. Security shortcomings and countermeasures for the sae j1939 commercial vehicle bus protocol. *IEEE Transactions on Vehicular Technology*, 67(5):4325–4339, 2018.

[13] Mustafa Ozcelikors and Akin Gumuskavak. Platform-independent infotainment and digital cluster development using yocto project. In *2020 International Conference on Electrical, Communication, and Computer Engineering (ICECCE)*, pages 1–5, 2020.

[14] Jan Schroeder, Christian Berger, Alessia Knauss, Harri Preenja, Mohammad Ali, Miroslaw Staron, and Thomas Herpel. Predicting and evaluating software model growth in the automotive industry. In *2017 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pages 584–593, 2017.

[15] Society of Automotive Engineers. J1587_201301: Electronic Data Interchange Between Microcomputer Systems in Heavy-Duty Vehicle Applications.

[16] Society of Automotive Engineers. J1708_201609: Serial Data Communications Between Microcomputer Systems in Heavy-Duty Vehicle Applications.

[17] Society of Automotive Engineers. J1939DA_202208: J1939 Digital Annex - SAE International.

[18] Society of Automotive Engineers. J2497_202311: Power line carrier communications for commercial vehicles.

[19] Society of Automotive Engineers. SAE J1939 Standards Collection. Accessed: 2021-11-09.

[20] SystemsCyber. Truck cape projects, 2022. Accessed: 2024-11-11.

[21] SystemsCyber. Ultimate truck hacking platform, 2024. Accessed: 2024-11-11.

[22] The Yocto Project. *Yocto Project Reference Manual*, 2024. `https://docs.yoctoproject.org/current/ref-manual/`.