Securing CAN Traffic on J1939 Networks

Jeremy Daily, David Nnaji and Ben Ettlinger
Colorado State University
{jeremy.daily,david.nnaji,ben.ettlinger}@colostate.edu

***Abstract*—Controller Area Network (CAN) implementations inherently trust all valid messages on the network. While this feature makes for easy replacement and repair of electronic control units (ECUs), this trust poses some cybersecurity challenges, like making it easy to spoof messages or alter them with a middle-person attack. With an SAE J1939 based network, the meaning of the network messages are often published, which reduces the amount of work needed to reverse engineer the protocol. Furthermore, J1939 is often used on high-value and high-risk cyber-physical systems, like trucks, buses, generator systems, construction, agriculture, forestry, and marine and military systems. Therefore, improving the cybersecurity posture of SAE J1939 networks is crucial for protecting critical infrastructure.**

The approach outlined in this paper for an intrusion detection system (IDS) uses so-called CAN Conditioners at or in each of the vehicle ECUSs that communicate with the Secure Gateway near the vehicle's diagnostic port. Each of the CAN Conditioners and the Secure Gateway includes an allowlist and blocklist procedure to prevent a variety of unauthorized network attacks. In addition, a cipher-based message authentication code (CMAC) is calculated by each node and transmitted across the network using the J1939 Data Security Message parameter group number (PGN). This CMAC message acts as a heartbeat indicator for the Secure Gateway to verify healthy node behavior and unaltered messaging.

Reference prototype hardware and software are described and results from a test implementation on a Class 6 truck with 6.7L diesel engine and an automated transmission are also described. The provisioning process sets up hardware security modules to be able to exchange secrets over the CAN bus using the elliptic-curve Diffie-Hellman protocol (ECDH). Once secrets are exchanged, ephemeral session keys are shared with the Secure Gateway, which keeps track of the CMACs from each CAN Conditioner. If a CMAC fails to match, the Secure Gateway informs the network using the J1939 Diagnostic Message #1 and a message using the J1939 defined Impostor PG Alert parameter group. Results show the IDS can detect alteration of a message or an impersonated message.

I. Introduction

Heavy vehicles are a vital part of the global economy as they are responsible for delivering people and goods to all corners of the world. These vehicles typically use SAE J1939 based networks. The J1939 network is built on top of the CAN2.0b specification and uses 29-bit identifiers. Not only do cybersecurity concerns with CAN based systems apply to J1939, but J1939 specific cybersecurity concerns also exist [1], [2]. In response to these concerns, the National Motor Freight Traffic Association reported on the state of heavy vehicle cybersecurity in a comprehensive whitepaper [3].

A. CAN and J1939 Cybersecurity Vulnerabilities

Together, the ISO 11898 standard and the SAE J1939 recommended practice define the common networking layers of the OSI reference model for heavy vehicle systems. As shown in Fig. 1, SAE J1939 exclusively defines the upper layers of the vehicle model. For example, SAE J1939 provides standardization for non-proprietary Arbitration IDs as well as multi-frame transport protocol messages.

OSI Reference Model	Vehicle Standards
Application	J1939/71 J1939/73
Presentation	Null
Session	
Transport	J1939/21
Network	J1939/31
Data Link	J1939/21 ISO 11898
Physical	J1939/11 J1939/12 J1939/14 J1939/15 ISO 11898

Fig. 1: Vehicle standards mapped to OSI Network Model.

The fundamental issue being addressed is that CAN messages are trusted by all receiving nodes on the bus. Every message received is treated as if it were already authorized. The CAN messaging protocol is a multi-master network, with all nodes receiving all messages. The only check performed on the message is a cyclic redundancy check (CRC) to validate the integrity of the frame field in the case of bit-flips. All messages are received by all nodes immediately upon sending. Individual modules do not include authentication systems. In SAE J1939 networks, the source address (SA), which is the last byte of the arbitration ID shown in Fig. 2, tells the network which controller application is sending the message. Since this value is self-reported, it can be falsified.

Priority	Reserved	Data Page	PDU Format	PDU Specific	SA
3 bits	1 bit	1 bit	8 bits	8bits	8 bits

Fig. 2: Structure of 29-bit Extended CAN ID.

Workshop on Automotive and Autonomous Vehicle Security (AutoSec) 2021
25 February 2021, Virtual
ISBN 1-891562-68-1
https://dx.doi.org/10.14722/autosec.2021.23031
www.ndss-symposium.org

This poses a cybersecurity risk because if a node becomes compromised, or if a rogue node is introduced to the network, the control units of the vehicle have no method of determining which messages are authorized and come from the proper source, and which messages are from the attacking node. Due to the historically air-gapped nature of vehicles, this vulnerability has not been a threat until recently, as any attacks would require physical access to the vehicle. Attacks based upon physical access to the system do not scale well. However, with the rise in wireless telematics devices, infotainment systems, and wireless electronic logging devices, new wireless attack surfaces have been introduced to the vehicle CAN network. These wireless attacks may be able to scale, which is a cybersecurity concern.

To combat the potential for cybersecurity attacks against heavy vehicle networks, an intrusion detection system is proposed that introduces traditional network cybersecurity to the legacy CAN protocol using PGNs for Data Security, Imposter PG Alert, and Diagnostic Messages as defined in the SAE Recommended Practice J1939 [4].

This is accomplished using devices at each of the network nodes and another on the diagnostic port. Each device implements a message send allowlist, which prevents its node from sending unauthorized messages, and an incoming message blocklist, which prevents fraudulent messages from reaching the protected node. To monitor the continued cybersecurity health of the devices, each device at the node transmit a message containing a CMAC, which is verified by the secure diagnostic node. Together, these measures mitigate the threat of compromised and rogue nodes on the network.

B. Background Information

Functionally, the CAN Conditioners and Secure Gateway devices utilize three useful security concepts: message authentication, firewalls, and intrusion detection. As noted earlier, message authentication is the process of determining if a message has been modified in transit or replayed. Message authentication can also extend to verification that the message came from a legitimate source. CAN uses message CRCs for fault detection but this is largely insufficient to prevent tampering. In modern Internet security, comprehensive authentication is usually accomplished with a combination of multiple techniques including checksums, message authentication codes (MACs), authenticated encryption, or digital signatures. However, constraints within CAN have made direct implementations of these techniques unwieldy. For example, the maximum data payload for a CAN frame is 8-bytes which limits the ability to append cryptographic information like a MAC since these often require at least twice as much data.

Despite this limitation, many message authentication approaches for CAN have been proposed. A review of secure communication approaches in 2018 identified and compared 8 unique methods for implementing MACs from various research publications [5]. For each method, the authors identified the type of algorithm used, the MAC position, MAC size, the computational requirements, and other constraints. CMAC was a commonly selected algorithm among the methods but had the highest computational cost. Despite the variety of methods reviewed, nearly all them lead to increased latency, increased bus load.

AUTOSAR SecOC also mentions CMAC for CAN frames but implementing it would require changes to the existing J1939 standard messaging [6]. Given a requirement for compatibility with legacy systems, approaches that require changes to the J1939 protocol hinder their adoption. Furthermore, many ECUs are required to operate under full bus loads and are resource-limited which prevents the adoption of computationally intensive approaches. However, research has suggested that the calculation times associated with CMACs could be sufficiently offset with the use of hardware-based acceleration [5].

To overcome the deficiencies of using MACs alone, a number of IDSs have been proposed. For example, engineers from NXP Semiconductors have proposed a distributed detection methodology that utilizes CAN message filtering at the transceiver level [7]. In short, the system leverages the NXP TJA115x CAN transceiver's supervisory capabilities to analyze received and transmitted messages of its host ECU in real-time. When faulty behavior is detected, the transceiver will reduce the functionality of its host by sending error frames to the bus [7].

Specifically, the transceiver can be provisioned with an allow and blocklist to prevent spoofing and tampering on the transmit and receive paths. It is also equipped with a message buffering system known as a "leaky-bucket" to minimize the threat of message flooding (DoS) attacks [7]. When these functions are combined into a security model, the result is analogous to a firewalls at each host ECU with added levels of authentication.

II. Needs Analysis and Conceptual Design

To determine the needs and propose a design concept, a threat model is needed. The scope of this project encompasses the internal J1939 heavy vehicle network and its connected ECUs. This means any diagnostic system, as described in [8], is out of scope. As a result, the following attack vectors based on the diagram in Fig. 3 are considered:

1) *Rogue Node*. A malicious node is installed onto J1939 network. The malicious node could be physically installed or the programming on an existing node could be compromised.
2) *Supply Chain*. A new node is installed that has malicious code pre-programmed.
3) *Middle-Person Attack*. A node is installed in the middle of the J1939 network and can alter a message from one side of the network to the other.

Any node on the vehicle can act in ways that violate the rules for normal transmission of messages. Therefore, the need is to detect the violations in a consistent and robust manner. Furthermore, system resilience is considered only up to violation detection as opposed to any additional mitigation.

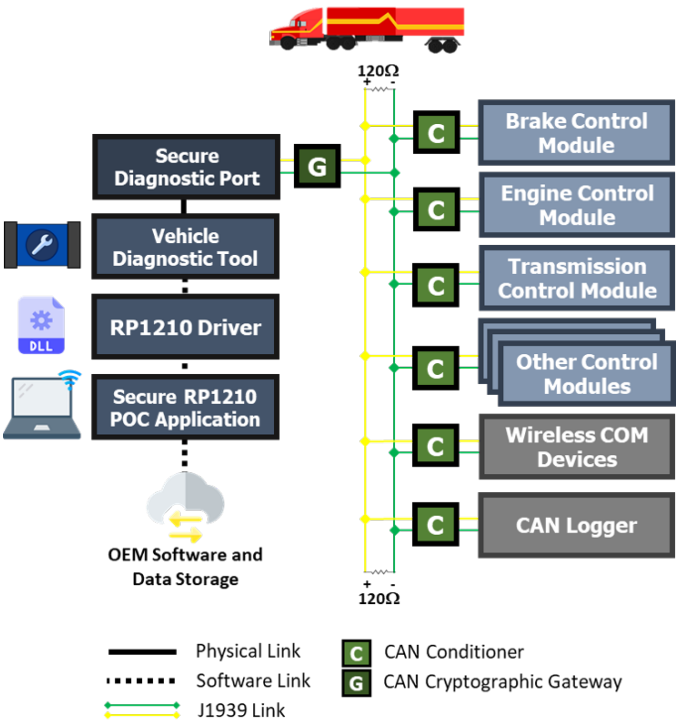

Fig. 3: J1939 network diagram with Secure Diagnostic Connection and Intrusion Detection System Installed

The concept shown in Fig. 3 aims to enforce the rules for the utilization of the CAN bus. The CAN Conditioners, denoted with a "C" are intended to watch and react to the network traffic coming in and out of each node. This concept was proposed by NXP with their "Stinger" CAN Transceiver [7]. This concept implements allowlist for known good messages, blocklists for messages known to not be allowed, and rate limiting to prevent bus flooding and enforce CAN timing.

The secure transceiver approach works only on IDs at this time. However, a middle-person attack can alter message content without detection. To detect message alteration, a message authentication code (MAC) must be calculated and periodically transmitted. Unfortunately, there are few practical ways to embed MACs with 8-byte J1939 frames. As such, the concept is to introduce a sentinel message that contains a MAC for comparison with the received MAC.

The Secure Gateway, denoted with a "G" in Fig. 3, collects all messages and calculates a MAC independent of the CAN Conditioner. The Secure Gateway will compare the MAC it calculates to the MAC it receives to determine if any contents have changed.

III. Requirements

There are two sets of requirements based on the needs of the project. The first set is for the heavy vehicle system and the second set is for cybersecurity. The vehicle system is depicted in Fig. 3. This is a single J1939 specified network connecting typical ECUs on a CAN bus. The system is compliant with SAE J1939 specifications, which means one or more unique controller applications (CAs) reside on an ECU. Each controller application is assigned a SA that is transmitted as the last eight bits in the 29-bit CAN identifier.

CAN Conditioners are installed in the stubs of the wiring harness in such a manner where only one ECU is present on the unprotected side and the J1939 backbone is connected on the other side. This makes the CAN Conditioner like a firewall or gateway device.

In a fashion similar to [9], we define our requirements in two categories, system and cybersecurity.

A. System Requirements

The system is a heavy vehicle using SAE J1939 for in-vehicle networking. The proposed modifications must satisfy the following requirements:

- SR1 CAN Conditioners must be able to support a throughput of 100% busload at 500kbps.
- SR2 Intrusions (i.e. rule violations) should always be detected.
- SR3 False Positives (i.e. detecting a violation when there was not) are not tolerated and should be less than 0.0001%, which is consistent with the reliability of other sensor systems on the vehicle.
- SR4 Longevity - the system must maintain its performance for long periods of operation (i.e. 24 hours) and long vehicle lifetimes (i.e. 20 years).
- SR5 The solution must be applicable to existing and legacy J1939 enabled vehicles. This means CAN frames are restricted to 8 bytes with 29-bit arbitration identifiers.
- SR6 The solution must comply with other vehicle system requirements.

Based on these system requirements, statistics or machine learning based intrusion detection systems are disqualified because the false-positive rate of those approaches is too high. Therefore, deterministic strategies based on message authentication codes are pursued instead.

Another restriction for the solution is the necessity to work with legacy vehicle systems. This eliminates the use of CAN-FD or Automotive Ethernet and the solution is restricted to use the CAN 2.0b specification of SAE J1939.

B. Cybersecurity Requirements

- CS1 Each device must have a unique, unknowable private key that is stored exclusively in a hardware security module (HSM).
- CS2 Randomly generated numbers must have high entropy and unpredictable seeds.
- CS3 No new ciphers or algorithms will be used for cryptographic operations (i.e. use well-known AES, ECC, and RSA ciphers)
- CS4 Session keys must never be transmitted in the clear over the network.

Based on these design requirements, the two components of the conceptual design will be discussed: (1) the Secure Gateway and (2) the CAN Conditioners.

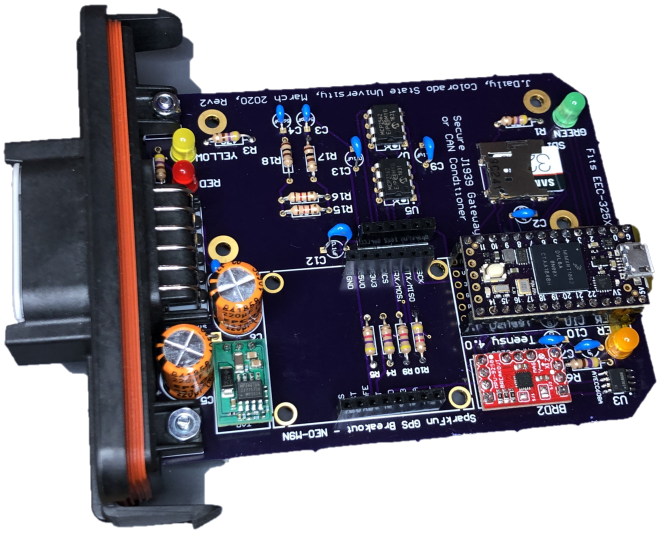

Fig. 4: Hardware prototype utilizing the Teensy 4.0 development board, the ATECC608A security module and 2 CAN transceivers.

IV. Secure Gateway

In [8], Daily and Kulkarni introduced a prototype secure gateway based on the Teensy 4.0 development board and the Microchip ATECC608A hardware security module. The paper detailed a strategy to provision the hardware security module and exchange device keys using the elliptic curve Diffie-Hellman (ECDH) protocol. Once device keys are exchanged, they are used to encipher session keys. These random session keys and initialization vectors are used to setup an AES-128 cipher in CBC mode. This cipher is used to encrypt and decrypt messages from a secure gateway (shown in Fig. 3) to a secure PC diagnostics application. However, the Secure Gateway can also communicate on the J1939 network, albeit not using enciphered traffic.

Since the hardware for the Secure Gateway has two CAN channels, a hardware security module, and a robust processor, it is selected as the hardware for both the CAN Conditioner and Secure Gateway. The detailed hardware prototype schematic is available on Github [10]. A prototype of the hardware is shown in Fig. 4.

V. CAN Conditioners

A. Functional Overview

As seen in Fig. 3, all incoming messages on the CAN bus and outgoing messages from an ECU must go through the CAN Conditioner (or Secure Gateway for diagnostic connections). The CAN Conditioner compares each incoming or outgoing message to the allowlist/blocklist to mitigate spoofing attacks. The allowlists and blocklists are based on the CAN ID, specifically the SA.

If an incoming message to a node, such as the Engine Control Module, contains a blocklisted SA, then the CAN Conditioner does not forward the message to the engine control module. This could be due to a rogue node sending fraudulent engine messages.

If an outgoing message from the protected ECU contains a SA which does not match the SA of its host listed on its allowlist, the CAN Conditioner does not forward the message to the rest of the J1939 network. This could be the case due to a compromised node sending fraudulent messages to the network.

Furthermore, each CAN Conditioner enforces a short pause between transmitting messages. This pause is on the order of 1-3 milliseconds, which is like the ”leaky bucket” defense.

B. CMAC Health Monitoring

To augment the protections provided by the CAN Conditioners, each Conditioner generates an individual shared secret with the Secure Gateway node. This shared secret is generated using a Diffie-Hellman key exchange, in which both devices share an encrypted session key, resulting in a unique session key known only to both devices performing the exchange. At all points during the exchange, the information is encrypted and indecipherable in transit. Even if another node captured all key exchange messages, the session key could not be learned due to the use of individual device private keys created when provisioning their HSMs.

Once a session key is securely generated and exchanged between a CAN Conditioner and the Secure Gateway, it is then used by the CAN Conditioner to generate a CMAC. A portion of this CMAC is transmitted across the network. Upon receiving a CMAC portion, the Secure Gateway generates its own CMAC based upon that CAN Conditioner’s unique session key. The same bytes of the CMAC generated by the gateway are compared with the received bytes of the CMAC generated on the CAN Conditioner. These two data portions should match, as both the CAN Conditioner and the Secure Gateway generated them using the same session key and nonce.

A new CMAC is then generated and sent once per second using portions of the previous message(s) in the calculation. This results in a unique CMAC being transmitted by the CAN Conditioner and checked by the Secure Gateway once per second. Each node performs these actions independently with the Secure Gateway.

There is a small chance that due to the timing of the CMAC generation, the two nonces used could be different, resulting in conflicting CMACs. However, assuming that is not the case in a CMAC mismatch, the offending node may be identified as malfunctioning or compromised, and this failure to authenticate is logged and further action may take place. If a node stops sending CMAC messages, something has interrupted the process, such as malfunction or a compromise to the device. Again, once detected, further action may be taken to mitigate the potential attack.

C. Secure Key Exchange Process

After the Secure Gateway and CAN Conditioner have been provisioned, connected to a J1939 network, and powered on,

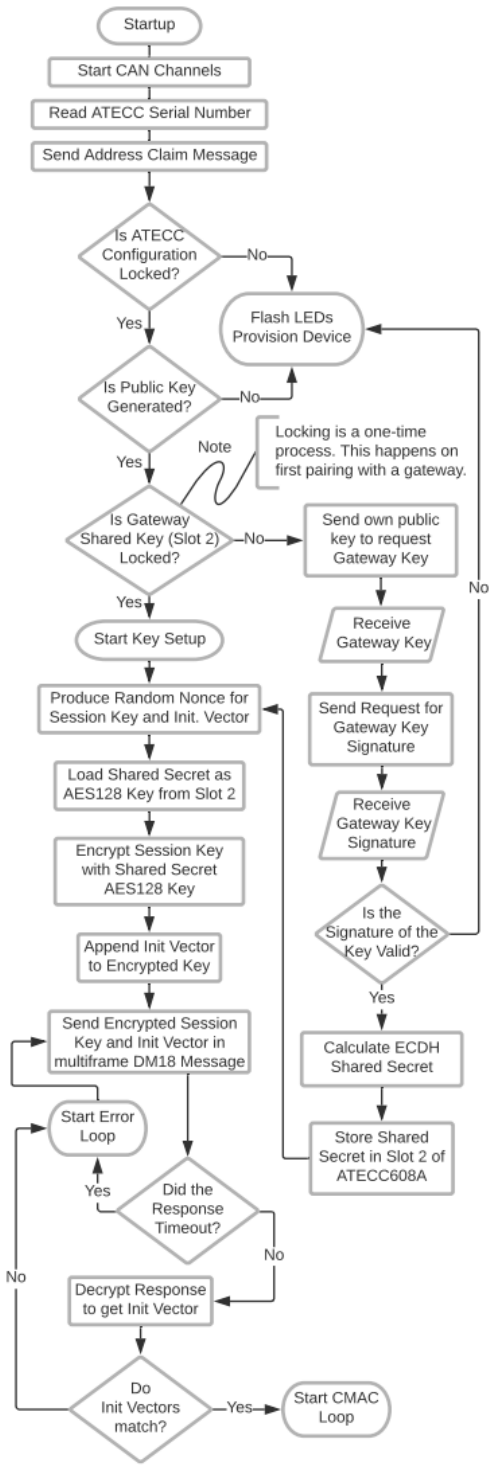

Fig. 5: Flow diagram of the CAN Conditioner startup sequence

the devices will generate their ECC key pairs. When this step is complete, a number of message exchanges take place to establish a secure session. Functionally, these processes are described in detail for the Secure Gateway in Fig. 8 and the CAN Conditioner in Fig. 5.

In sum, the setup of a secure session begins after both devices have claimed an address and the CAN Conditioner sends its own public key. These messages are broadly facilitated by the transport protocol described in SAE J1939-21 which allows for the transmission of multi-frame messages. It is also facilitated by the data security PGN, DM18, defined in SAE J1939-73, which is used to identify cryptography procedures and information.

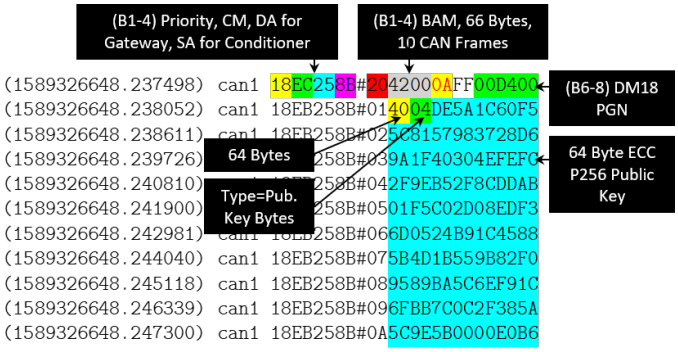

Fig. 6: CAN Conditioner sends its own public key with J1939 transport protocol.

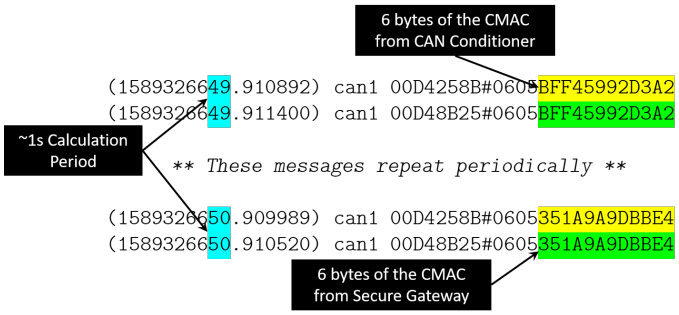

Fig. 7: Periodic CAN Conditioner CMAC messages and Secure Gateway CMAC confirmation responses.

An example transmission of the CAN Conditioner public key is shown in Fig. 6. The first message includes the transport message type (CM), destination address for the gateway (0x25), SA of the conditioner (0x8B), the number of segments needed for the rest of the message (10), and the security PGN (0x00D400). The following messages include the segment number, the number of bytes of information (64 bytes), and the key type (public ECC P256). The CAN Conditioner will repeat this message periodically until it receives a response from the Secure Gateway.

When the CAN Conditioner public key is received, the Secure Gateway responds in a similar fashion with its own public key. After both public keys have been exchanged the CAN Conditioner generates a 16-byte AES-128 session key and 10-byte initialization vector using the HSM and sends the pair.

When the Secure Gateway receives the data pair, it calculates an AES encrypted seed and sends it to the CAN Conditioner. If the CAN Conditioner deciphers the encrypted seed and it matches what it sent, then the session is established.

After a secure session is established, the CMAC values are calculated and sent between the two devices. As shown in Fig. 7, these exchanges repeat roughly once per second.

The key features in this process are as follows:

1) Private keys are never known or revealed since they exist in the HSM.
2) Pre-shared secrets are calculated by the HSM but never transmitted.
3) Envelope encryption takes place in the HSM.
4) Session keys are ephemeral and securely shared.
5) Public key exchange only needs to take place upon initial installation.

VI. Test and Evaluation

The Secure Gateway system has two internal counters tracking the number of times the CMAC calculations match and the number of times the CMAC calculations do not match. With these counters, and a controlled experiment, the false positive and false negative rates can be estimated. A false positive is when a CMAC calculation fails to match even though there were no attacks. A false negative is when there was an attack, but the CMAC comparison indicated no messages were changed.

Each SA has its own counter. This means if a node impersonates a controller application that already has a CMAC setup, then the messages using the same SAs will be included in the CMAC calculation on the Secure Gateway. Since the impersonated messages are not used in the CAN Conditioner CMAC (implying they didn't originate from the correct ECU), the CMAC comparison will fail and the system would report a true positive. This feature worked in all tests.

As expected, every time a message was altered, the CMAC calculations did not match. This is an expected result since the false negative rate is on the order of the collision rates in an AES-128 based CMAC, which is acceptably low. Therefore, the requirement for low false negatives is met.

The false positive rate is determined by a ratio of the reported CMAC mismatches to the number of CMACs calculated with no intrusions. Due to the way CMACs are calculated, the timing of the sentinel message containing the CMAC from the CAN Conditioner must provide sufficient gaps such that the same messages are used to calculate the CMAC on the Secure Gateway. If the order of a message is changed, then the CMAC comparison would fail, leading to a false positive. After some initial tuning of the timing, a 20 millisecond gap on either side of the sentinel message lead to a low false positive rate. In one bench test that went for 24 hours, there were 508,762 CMACs that matched and 12 that did not. This is a false positive rate of 0.00236%, which is arguably too much. This would lead to false maintenance actions and operators would tend to ignore the warnings. However, this is a promising preliminary result and similar numbers were found when testing on vehicle. Additional logic regarding the number of messages and better tuning of the timing parameters should drop the false positive rate even lower.

VII. Conclusion

Correcting the security vulnerabilities caused by the inherent trust among devices connected by CAN is an ongoing security challenge among researchers. This paper described a legacy system IDS for CAN2.0b-based systems that aimed to provide stronger authentication of legitimate devices, detect the occurrence of system compromise and minimize the impact of a network attack. This was accomplished through the use of CAN Conditioners and a Secure Gateway device to segment network nodes and verify healthy node behavior. Based on our experiments, this concept shows promising preliminary results towards mitigating rogue nodes, supply chain, and middle-person attacks while minimizing network resource consumption typically associated with cryptography-based approaches. However, further work is needed to reduce the hardware for practical use. With further development, it has the potential to act as a mechanism for enhancing vehicle security.

Acknowledgment

This work is partially supported by the Defense Automotive Technologies Consortium (DATC) through DG Technologies under Award DA2-PA027C. Also partial support from Colorado State Bill 18-086 is gratefully acknowledged.

References

[1] S. Mukherjee, H. Shirazi, I. Ray, J. Daily, R. Gamble, "Practical DoS Attacks on Embedded Networks in Commercial Vehicles," *Int. Conf. Information Systems Security.*, 2016, pp 23-42, doi:10.1007/978-3-319-49806-5_2

[2] Y. Burakova, B. Hass L. Millar, and A. Weimerskirch, (2016) "Truck Hacking: An Experimental Analysis of the SAE J1939 Standard," 10nth USENIX Workshop on Offensive Technologies, [Online]. Available: www.usenix.org/conference/woot16/workshop-program/presentation/burakova

[3] National Motor Freight Traffic Association, Inc., (2015) "A Survey of Heavy Vehicle Cyber Security," Nat. Motor Freight Traffic Association, [Online]. Available: www.nmfta.org/documents/hvcs/nmfta heavy duty vehicle cyber security whitepaper v1.0.3.6.pdf

[4] *J1939 Digital Annex*, SAE Standard No. J1939DA_202001, Jan. 2020.

[5] Q. Hu and F. Lou. (2018). "Review Of Secure Communication Approaches For In-Vehicle Network," Int. Journal of Automotive Technology, 2018 pp. 879-894, doi:10.1007/s1223901800851

[6] *Specification of Secure Onboard Communication*, 654, AUTOSAR, 2017.

[7] Houck, A., "Secure CAN Transceiver," NXP, AMF-AUT-T2854, 2018, [Online]. Available: community.nxp.com/t5/Technology-Days-Training/Secure-CAN-Communication-The-System-Impact-by-Software-Based/ta-p/1104506

[8] J. Daily, P. Kulkarni, "Secure Heavy Vehicle Diagnostics," In *Proc. of the Ground Vehicle Systems Engineering and Technology Symposium (GVSETS)*, NDIA, Novi, MI, Aug. 13-15, 2020.

[9] Ammar, M., Janjua, H., Thangarajan, A., Crispo, B. et al., "Securing the On-Board Diagnostics Port (OBD-II) in Vehicles," SAE Int. J. Transp. Cyber. & Privacy, pp. 83-106, 2019, doi: 10.4271/11-02-02-0009

[10] J. Daily "Secure J1939 Gateway," Last Accessed 11 Jan 2021. [Online]. Available: github.com/SystemsCyber/CANWatermarking

[11] *Electronic Logging Devices and Hours of Service Supporting Documents*, 78292, FMCSA–2010–0167, Department of Transportation, Federal Register, 2015. [Online]. Available: www.govinfo.gov/content/pkg/FR-2015-12-16/pdf/2015-31336.pdf

[12] S. Mukherjee, J. Walker, I. Ray and J. Daily, "A Precedence Graph-Based Approach to Detect Message Injection Attacks in J1939 Based Networks," 2017 15th Annual Conference on Privacy, Security and Trust (PST), Calgary, AB, 2017, pp. 67-6709, doi: 10.1109/PST.2017.00018.

Fig. 8: Flow diagram for the Secure Gateway processing J1939 messages