The Advantages of Distributed TCAM Firewalls in Automotive Real-Time Switched Zonal Networks

Evan Allen
Virginia Tech
evanallen@vt.edu

Zeb Bowden
Virginia Tech
Transportation Institute
zbowden@vt.edu

J. Scot Ransbottom
Virginia Tech
ransbottom@vt.edu

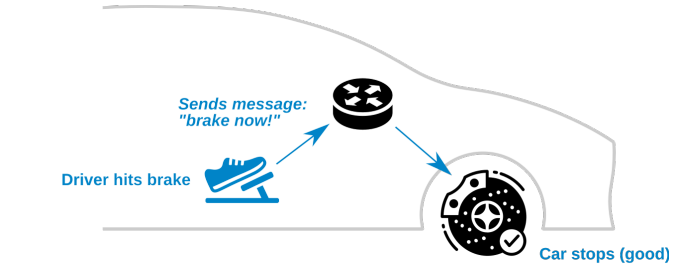
***Abstract*—Attackers have found numerous vulnerabilities in the Electronic Control Units (ECUs) of modern vehicles, enabling them to stop the car, control its brakes, and take other potentially disruptive actions. Many of these attacks were possible because the vehicles had insecure In-Vehicle Networks (IVNs), where ECUs could send any message to each other. For example, an attacker who compromised an infotainment ECU might be able to send a braking message to a wheel. In this work, we introduce a scheme based on distributed firewalls to block these unauthorized messages according to a set "security policy" defining what transmissions each ECU should be able to send and receive. We leverage the topology of new switched, zonal networks to authenticate messages without cryptography, using Ternary Content Addressable Memory (TCAMs) to enforce the policy at wire-speed. Crucially, our approach minimizes the security burden on edge ECUs and places control in a set of hardened zonal gateways. Through an OMNeT++ simulation of a zonal IVN, we demonstrate that our scheme has much lower overhead than modern cryptography-based approaches and allows for real-time, low-latency (<0.1 ms) traffic.**

I. Introduction

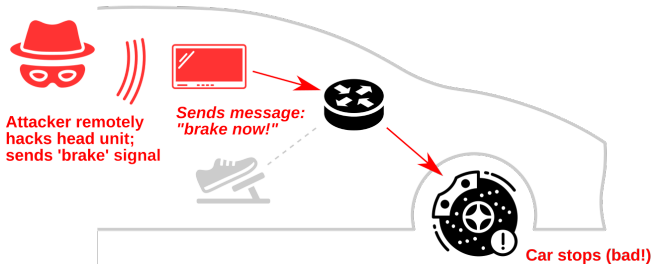
The past decade has seen the rise of vehicle cyberattacks that enable a malicious user to take control of vehicles by exploiting the network of Electronic Control Units (ECUs) inside them.

In 2015, *Wired* magazine revealed that a vulnerability in the OnStar ECU of certain General Motors vehicles made it possible to remotely disable the vehicle's brakes at high speeds [15]. Cybersecurity experts and noted hackers Miller and Valasek were able to remotely cut the engine of a Jeep Cherokee over the cellular network [26] and found ways to remotely control its steering and acceleration the next year [16]. Keen Security Lab in China found vulnerabilities in a Tesla two years in a row, allowing them to remotely control the brakes [39]. More recently, in 2023, hackers at the Pwn2Own competition found a remote vulnerability in a Tesla's Bluetooth module [32].

These attacks often follow a pattern: hackers find a vulnerability that allows them to control one ECU, and then they use that ECU to transmit malicious messages to other ECUs that can perform dangerous actions such as unexpected steering or braking. In conventional IT security, this practice of using one compromised system in a network to attack other parts of a network is called *pivoting* [19].



(a) Normal scenario. Driver chooses to brake.



(b) Malicious scenario. Attacker brakes car against driver's will.

Fig. 1. Normal vs. malicious scenarios.

Pivoting often exposes an attacker to a variety of important internal ECUs that they could not directly access before, giving the attacker more options for controlling the vehicle. Consider the scenarios in Figure 1. Scenario (a) illustrates the normal case where a driver chooses to brake, sending a message from the braking pedal to the wheels. We are concerned with scenario (b), where an attacker has remotely compromised an ECU such as the head unit and wishes to stop the car. The head unit normally handles the infotainment system and is not directly responsible for controlling the vehicle. However, the attacker may abuse this ECU to send malicious messages to the rest of the vehicle, causing it to brake or shut off against the driver's will.

In theory, this attack should not be possible. While the head unit may need to receive control messages such as speed readings for the user, it should never be able to send them,

Symposium on Vehicles Security and Privacy (VehicleSec) 2024
26 February 2024, San Diego, CA, USA
ISBN 979-8-9894372-7-6
https://dx.doi.org/10.14722/vehiclesec.2024.23012
www.ndss-symposium.org

and other ECUs should never be able to receive them. The attacker's messages should not just be considered malicious—they should be *invalid*.

If there was some way to limit each ECU so that it could only send and receive message types that it "should" send/receive according to an automaker's design, then attackers would have dramatically fewer options after compromising a single ECU. In our example, the attacker would not be able to send control messages because the head unit would not be designed to send them.

A. In-Vehicle Network (IVN) Security Policies

Formally, such a security measure would enforce a *security policy* $\mathcal{S}$ consisting of a set of 3-tuples in the form

$$\{(\text{sender } s,\ \text{msg\_type } m,\ \text{receiver } r), ...\}$$

that describes which ECUs can send which message types to which other ECUs. Any message that satisfies one of the tuples is considered *authorized* and should be allowed. All other messages are considered *unauthorized* and should be blocked.

In Figure 1, one such authorized tuple in the security policy might be (`brake_pedal`, `brake_msg`, `wheel`) to allow the driver to brake. This means that the brake pedal is allowed to send a brake message to the wheel. The attacker's unauthorized message (`head_unit`, `brake_msg`, `wheel`) would not be in the security policy, and thus should be blocked.

B. Enforcement

Many current IVNs do not provide adequate security measures to enforce these policies and prevent "unauthorized" messages from reaching their destination [3], [12]. When they arrive, receiving ECUs often have no way to know the true sender of the messages or if their senders were authorized to transmit those messages.

Techniques to enforce IVN security policies and stop such attacks do exist in modern literature, but they are often difficult to implement. Any such security solution must solve the following fundamental problems:

1) *Authentication.* For a given message, who sent it (s)? The message type m and receiver r can be found from inspecting a message, but an attacker could lie and supply a false value for the sender s.
2) *Authorization.* For a given message, was it allowed to be sent/received by the ECUs involved? (i.e., is (s, m, r) for this message in the security policy)?

Most current research focuses on using cryptography to solve part (1), authentication, with the implicit assumption that the receiving ECU will do the authorization and knows which sending ECUs are allowed to send it which messages (i.e., the policy). For example, many researchers attempt to mitigate *message spoofing* attacks [3], [12] where an attacker lies about the sender s.

Developing these solutions for IVNs is difficult because (a) IVNs have very stringent performance requirements for the latency and throughput of their data [1], [8], [18], leaving little room for security computations, and (b) it is difficult to standardize a new security protocol and persuade each ECU supplier to implement that protocol. This makes traditional, cryptography-based schemes more costly than they are in enterprise networks where computation time is more abundant and the stakes are not safety-critical.

In addition, it is important for security schemes not to ignore part (2), authorization. How would the security policy be managed? How would it be distributed to the ECUs that must decide whether the sender of an (authenticated) message is a valid one? Leaving authorization to the receiving ECUs creates many administrative challenges.

However, emerging IVN technologies such as the zonal architecture, which groups ECUs together based on location, and Automotive Ethernet (AE), which can connect ECUs in a switched network, introduce new possibilities for enforcing and managing IVN security policies using distributed firewalls at the network level.

In this paper, we investigate ways to take advantage of switched, zonal IVNs to ensure that ECUs can only send authorized types of messages while adhering to vehicle performance requirements and imposing minimal burden on the ECUs themselves. We introduce a distributed firewall that uses Ternary Content Addressable Memory (TCAM) to efficiently enforce a security policy in a way that is easier for automakers to control.

II. Threat Model

This work considers a zonal AE network and seeks to protect it from a remote attacker that has compromised one of the ECUs in the IVN. We do this by enforcing a security policy $\mathcal{S} = \{(s, m, r), ...\}$ as defined in Section I-A.

a) Attacker goals: The attacker wishes to accomplish a malicious goal G such as stopping the vehicle. The attacker has compromised some ECU C, and intends to send a message m to a target ECU T that is capable of accomplishing G. As a tuple, this message is (C, m, T). We assume that

1) $C \neq T$ (i.e., the compromised ECU C is not directly capable of accomplishing the malicious goal G and must send a message to a different ECU T to do so);
2) (C, m, T) is not in the security policy $\mathcal{S}$ (i.e., C would never normally send the message m that would accomplish G, so it is an unauthorized message).
3) The network gateways are trusted. While we recognize this may not always be the case (e.g., supply-chain attacks), we find this a realistic and necessary assumption to make. See Section VIII for more justification.

b) Defender goals: We wish to enforce the policy $\mathcal{S}$ and prevent the attacker from accomplishing their goal G. In other words, we wish to block the unauthorized message (C, m, T).

Consider the second scenario in Figure 1. In this case, the malicious goal G is to stop the vehicle, the compromised node C is the head unit, the target node T is a wheel, and the message m is a braking message. The tuple (C, m, T) is unauthorized here, because the head unit should never send a braking message to a wheel.

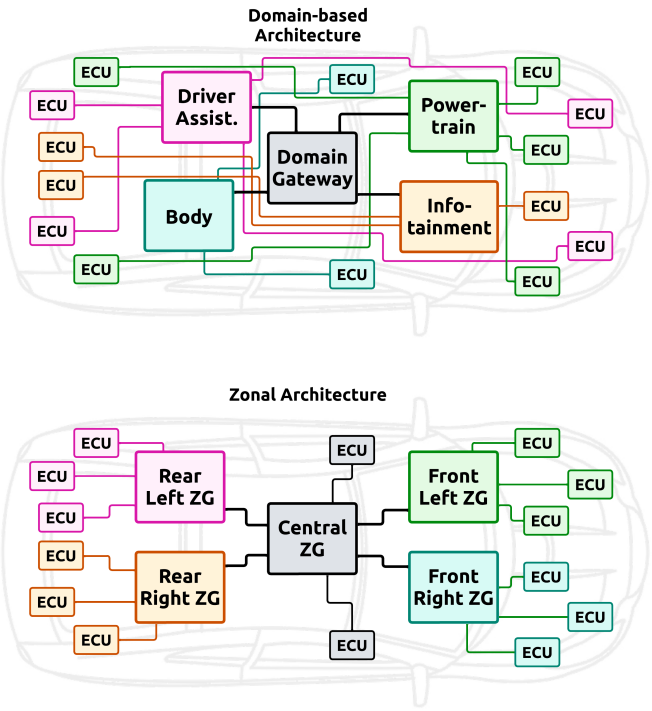


Fig. 2. Automotive IVN architectures.

III. IVN Background

Automotive IVNs consist of a variety of ECUs that are connected throughout a vehicle. ECUs are responsible for controlling various aspects of the car, such as the engine, power steering, entertainment system, and more.

Modern IVNs are largely based on bus technologies such as the Controller Area Network (CAN), FlexRay, and the Local Interconnect Network (LIN) [3]. These bus IVNs connect multiple ECUs together on a single wire, meaning that every message is a broadcast message. Bus topologies are popular because they reduce wiring complexity and weight [8].

A. Security Challenges with Buses

However, bus topologies face significant security challenges because of their nature as broadcast networks. ECUs that receive a message cannot know the true sender without additional work because any ECU on the wire could have sent it. While some protocols such as CAN include an ID field that specifies each message's type, neither CAN, LIN, nor FlexRay provide authentication features. Many groups have proposed ways to add authentication to IVNs such as [10], [11], [18], [25], [33], [37], [38], but these often require cryptographic computation that can cause unacceptable overhead [1], [18] (see Section III-D).

In addition, bus networks connect ECUs directly to each other. This means that it is difficult to create a security entity that can detect and block unsafe messages before a receiver ECU processes them. Thus, bus topologies place the security burden on the ECUs themselves, increasing the computational load on each (often under-equipped) ECU and requiring suppliers to agree on a new security standard.

B. Automotive Ethernet

Automakers have recently begun to use AE [21], an IVN technology based on conventional Ethernet [21], because it achieves much higher bandwidths while providing Time-Sensitive Networking (TSN) [20] features for low-latency communication. The most common types of AE are 100BASE-T1 and 1000BASE-T1, which can transmit data in full-duplex at 100 Mbps and 1 Gbps, respectively [8]. These forms of AE provide one-to-one connections between nodes and can be used to form a switched network like those found in conventional Ethernet.

C. Network Architectures

Most IVNs originally used a distributed architecture where each ECU was connected on a set of buses to a single, central gateway [28], [34]. However, IVNs have begun to evolve over recent years.

1) Domain-based Architectures: Due to rising data transmission and processing requirements, many automakers adopted a domain-based architecture that groups ECUs by their function (see Figure 2, top). For example, infotainment ECUs such as a head unit, speakers, and antenna might be connected in an infotainment domain, while powertrain ECUs such as the engine might be connected in a separate powertrain domain. Domains consist of a single "domain controller" that connects to the domain's ECUs and can forward messages back and forth to a central domain gateway [28], [31].

However, domain-based architectures are expensive to implement in real vehicles because they require significantly more wiring [14], [28]. This is because ECUs in the same domain may be distributed far away from each other (such as cameras in the front and back of the car), requiring each domain's wiring harness to separately cover the entire vehicle. The extra wiring complexity increases the weight of the vehicle and makes maintenance more difficult.

2) Zonal Architectures: Many automakers are moving towards a zonal architecture, which groups ECUs together based on their physical location instead of their function (see Figure 2, bottom). For example, a "front left" zone might include a wheel speed controller, camera, and ultrasonic sensor that all exist in the front-left part of the vehicle. A zonal gateway (ZG) connects ECUs in a zone and routes traffic to other zones along a higher-speed backbone link.

Zonal architectures provide significant cost advantages over domain-based architectures because they require less wiring and are easier to scale [2], [14], [28], [34]. In this paper, we focus on zonal IVNs because of their increasing popularity.

D. Performance Requirements

IVNs must be able to satisfy a wide range of performance requirements. Certain control data traffic (CDT) must have an end-to-end (E2E) latency of less than 0.1 ms, while cameras and other sensors demand throughputs of 40 Mbps and more [1], [8]. In a zonal network, the ZGs and backbone connections must handle multiple of these streams simultaneously.

These requirements make it difficult to impose normal security measures on the IVN, since adding even light cryptography elements to resource-constrained ECUs significantly bottlenecks throughput and raises latency. Hu, et al. [18] benchmarked a variety of symmetric cipher and hash suites on a microcontroller similar to many ECUs. They found that they could not exceed 15 Mbps throughput with any of the suites they tested, which included ChaCha20-Poly1305 [27], various AES-128 modes, and a basic SHA-256 hash. In [1], Allen et al. investigated various encryption-based authentication schemes for AE IVNs and found that none of them could satisfy both the ultra-low latency and high throughput requirements of modern IVNs without hardware acceleration, which would be expensive to put on each ECU.

E. Ternary Content Addressable Memory (TCAM)

To meet network requirements like these, many switches possess TCAM modules that allow the switches to forward and filter packets at wire speed. TCAMs operate differently than standard Random Access Memory; instead of allowing a user to find data based on a location or address, TCAMs allow a user to find data based on the *contents* of the data itself. TCAMs are organized as a set of key-value pairs where a user can look up a key and get the associated value in a single clock cycle. For example, a switch can search a TCAM table using the destination address of a packet to find which interface to send it on [40].

TCAMs are considered "ternary" because their key entries are strings of bits that can each take three possible values: 0, 1, or X (don't care). This allows for very flexible key matching rules, such as matching any of the searches 01100, 01101, 01110, and 01111 from a fixed key 011XX [9].

TCAMs are popular in switching applications because of their speed, but they can be expensive due to their high power cost. In addition, many TCAMs are only able to store a few hundred to a few thousand entries, which can make large-scale use difficult [9]. Still, many semiconductor companies in the automotive field use TCAMs, including NXP, Marvell, and Broadcom [40].

IV. Related Work and Current Approaches

There have been many efforts to build a security scheme that enforces an IVN security policy, though most are written to prevent specific attacks such as message spoofing and ECU impersonation, as documented in surveys like [12] and [3]. Many also focus on additional goals such as message confidentiality, but we focus only on the aspects relevant to our threat model.

A. Encryption-based Approaches

Most recent approaches have used some form of cryptographic authentication to prevent message spoofing, either by encrypting entire messages or attaching Message Authentication Codes (MACs) that verify the sender of a message. The main assumption is that receiving ECUs will authenticate the sender of a message and discard it if they do not expect that message from that sender. Most of the papers briefly recapped below try Hash-based MACs (HMACs) that utilize a hash function such as MD5, SHA1, or SHA256 to create MACs.

Yang et al. [10] proposed using the symmetric cipher AES-128 [13] and the MAC algorithm HMAC-SHA1 to authenticate messages, testing their scheme on a video stream with the CANoe.Ethernet development platform [36]. They reported that it was fast enough to play the video successfully, but they did not specify the video's datarate or focus on the E2E latency of the stream. Li et al. [11] continued this research by introducing an improved version of AES-128 and an MD5 authentication algorithm for AE, finding that their version was faster than standard AES-128/MD5. These papers showed promise for the speed of authenticated encryption in IVNs, but they did not investigate the schemes' latency overheads or their effects on safety-critical traffic.

Ma et al. [25] proposed a solution for a domain-based architecture that protects communications between the domain controllers. The domain controllers coordinate with a central key management center module that provides them temporary session keys. To communicate with each other, each domain controller attaches a MAC to its messages based on authenticated encryption schemes such as AES-256-GCM [13] or ChaCha20-Poly1305. Ma et al. found that their system only added about 0.7 ms overhead for an 8-byte packet using the SOME/IP application protocol [5], which is low but still does not satisfy the 0.1 ms requirement of CDT. Their scheme is resistant to many malicious attacks between domain controllers, but does not prevent message spoofing between ECUs within the same domain and requires adding a new ECU dedicated to security.

Silva et al. [33] evaluated the E2E delay of various symmetric AES/HMAC algorithms in the context of automotive, safety-critical communication. They ran these benchmarks on a pair of Tiva C Series microcontrollers with hardware cryptography modules and Real-Time Operating Systems. An off-the-shelf switch connected the microcontrollers. Silva et al. found that they could achieve E2E delays of about 0.3 ms for 64-byte packets with only about 0.1 ms of encryption overhead.

Wang et al. [37] devised an AE security protocol based on DES and HMAC-MD5 that had very low authentication overhead (around 0.1 ms), but relied on all ECUs sharing the same symmetric key. This makes it difficult for an attacker to send messages from a new node, but does not protect against an attacker who has already compromised an existing ECU.

Finally, Hu et al. [18] designed Gatekeeper, an AE protocol where a central switch or router acts as an authenticator and verifies the sender of each packet using HMAC-SHA256 tags. Gatekeeper's latency overhead is small with a single receiving ECU but increases with more ECUs. Hu et al. also struggled to achieve high throughputs on their test microcontroller, as explained in Section III-D.

B. Firewall-based Approaches

Other researchers have investigated the feasibility of using firewalls to block unauthorized messages in AE IVNs. In contrast to approaches based on authentication, firewall-based approaches attempt to quickly identify malicious messages based on a set of rules and drop them before they can reach their destination.

One of the first papers on an AE firewall was Pesé and Schmidt's [29] work, where they proposed a hardware/software

codesign to fit automotive constraints. Their firewall consisted of a Field-Programmable Gate Array that quickly compared packets against a simple whitelist with a microcontroller that checked more complex filtering rules in software. They evaluated their firewall in the context of a domain-based architecture and placed it between a domain controller and the central domain gateway. They found that their throughput was limited only to around 2–3 Mbps, and their E2E latency increased at about $\sim 4\,\mu s$ per software filter rule. They did not use TCAMs.

Luo and Hou [24] evaluated another AE firewall using microcontrollers and software packet filtering. They found that their system operated at 200 Ethernet packets per second with about $500\,\mu s$ latency, which is even slower than Pesé and Schmidt's result.

Lastly, Yilmaz [40] created a model of a combined firewall and intrusion detection and prevention system (IDPS) for a domain-based AE architecture. Yilmaz focuses on how to implement firewalls that use TCAMs to examine the header bytes of packets at wire-speed, allowing for much better performance than a simple software filter.

Automakers are also starting to incorporate AE firewalls. For example, ESCRYPT's CycurGATE router includes a combined hardware/software AE firewall that can allegedly process most packets at wire-speed [17]. Tesla also uses AE firewalls in its vehicles on a central Ethernet switch, which is normally a Marvell 88ea6321. This firewall uses a TCAM to block packets that attempt to spoof their IPs as certain important internal ECUs [6].

Firewall-based approaches are attractive because they move some of the security burden away from the ECUs and centralize it in a few higher-cost gateways. This reduces the cost of ECUs, removes much of the administrative complexity of asking each ECU manufacturer to implement a new cryptographic protocol, and puts more control into the hands of the automaker that builds the network. However, the above research suggests that firewalls can only process packets at wire-speed when implemented in hardware with technologies like TCAMs, which are difficult to write complex rules for.

V. Proposed Design

We claim that a switched, zonal architecture makes firewalls much more viable in an IVN, however, and propose a system of distributed TCAM firewalls that restrict each ECU's traffic to only what is authorized under the policy $\mathcal{S}$.

Consider the diagram of a zonal IVN in Figure 2. We propose placing firewalls in each ZG to enforce the security policy for the ECUs in that zone. For this paper, we assume that all connections in our zonal IVN are AE, with each message's type listed as a four byte integer somewhere within the first few bytes of the AE packet payload. This is common for message protocols such as CAN over Ethernet [30].

A. Authentication-by-position

Like any scheme to enforce the policy $\mathcal{S}$, we must somehow authenticate messages (determine their true sender s) so we can decide if they are authorized. Previous bus-based IVN technologies such as CAN or FlexRay connected many nodes together into a single broadcast domain, making it difficult for nodes to know the true sender of an incoming message.

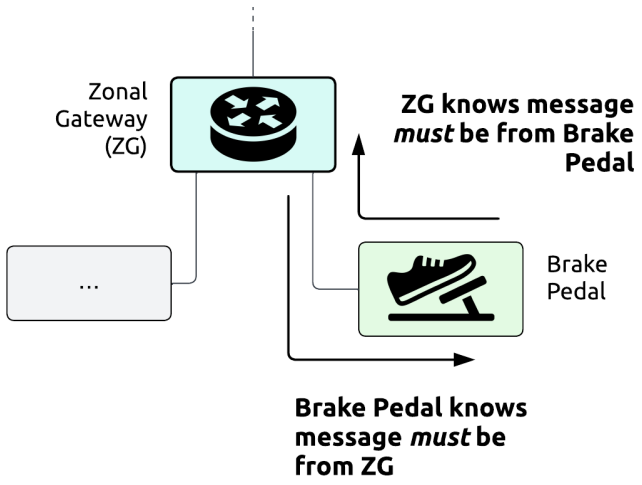


Fig. 3. Authentication-by-position.

However, most AE types (100BASE-T1, 1000BASE-T1) are designed for switched networks with one-to-one connections. In a zonal topology, this means that ECUs in a given zone are connected to a zonal controller on a dedicated physical port. This architecture, shown in Figure 3, provides a variety of advantages.

1) Each ZG knows the true source of any message originating from its zone based on the interface it entered the gateway on.
2) Each ECU knows any message it receives must have passed through its respective ZG.
3) If the ECUs trust the ZG, then they can offload their security burden to the ZGs.

Thus, the ZGs effectively authenticate messages from their own zones *by default* without requiring any cryptographic overhead.

B. ZGs Enforce Security Policy with TCAMs

Since the ZGs are trusted, they form a *trusted backbone* that connects the (possibly compromised) ECUs in each zone. Every message enters the trusted backbone when it arrives at the sender's ZG, which means that every message is authenticated by default. This gives the ZGs all the information they need about each message (s, m, r) to enforce the security policy $\mathcal{S}$. When a ZG receives a message from one of the ECUs in its zone, it checks the following things:

1) Does s match the listed sender for that interface (authentication)?
2) Is the message authorized, i.e., is $(s, m, r) \in \mathcal{S}$ (authorization)?

The ZG only forwards the message if it passes each check, blocking messages with a spoofed sender s or an otherwise unauthorized type/receiver. Figure 4 illustrates how our design blocks spoofed or unauthorized messages and allows authorized ones:

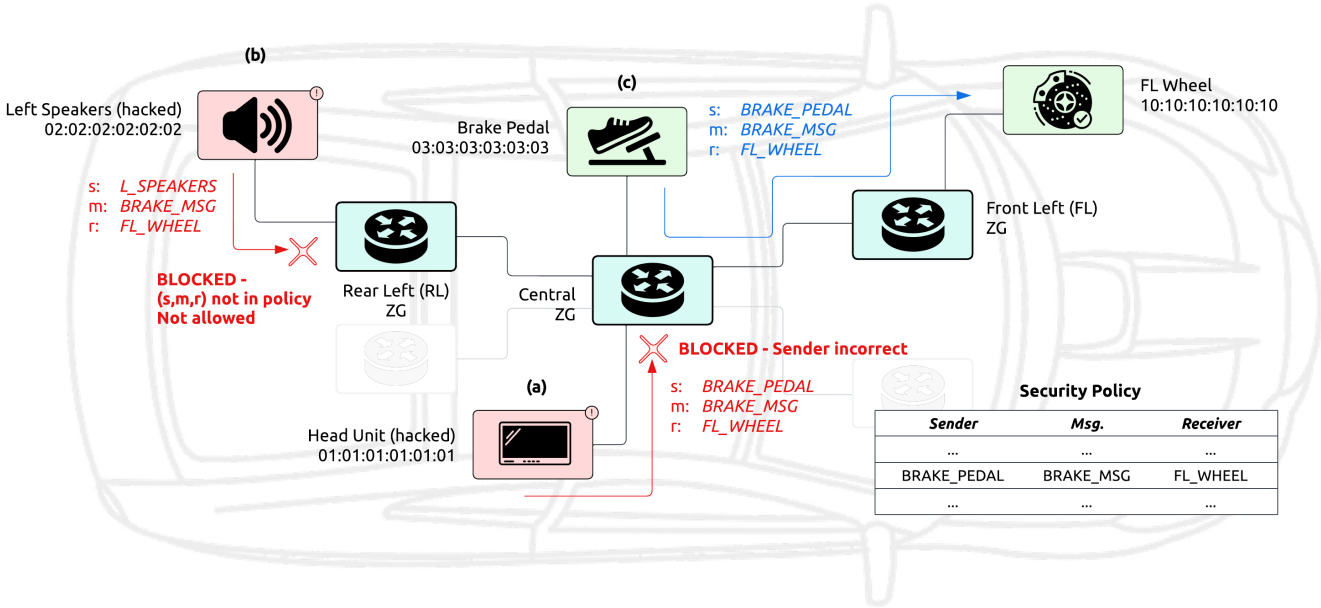


Fig. 4. ZG firewall blocks spoofed or unauthorized messages. (a) Message spoofing. (b) Unauthorized message. (c) Legitimate message.

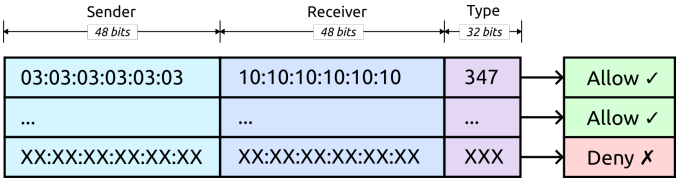


Fig. 5. Example TCAM table for the Central ZG's top interface in Figure 4.

a) Message spoofing: A hacked head unit *spoofs* a message as if it is the brake pedal, but the nearest ZG notices the listed sender `BRAKE_PEDAL` does not equal the one registered for that interface (`HEAD_UNIT`) and blocks it.

b) Unauthorized message: A hacked speaker sends an unauthorized braking message. The nearest ZG determines that this message (`L_SPEAKERS, BRAKE_MSG, FL_WHEEL`) is not allowed by the policy and blocks it.

c) Legitimate message: The braking pedal sends a braking message. The nearest ZG validates that the listed sender is correct and that the message is authorized according to the policy, then forwards it.

Switches can implement this with TCAMs to achieve wire-speed processing with fixed, nanosecond-scale delays [35]. Assuming s and r are represented as 48-bit Media Access Control addresses, and m is represented as a 32-bit message type field, TCAM entries to enforce $\mathcal{S}$ must be at least $2 \cdot 48 + 32 = 128$ bits long. While switches vary in TCAM capabilities, many have TCAMs that support entries of 134 bits or more and support user-defined fields for extracting the message type deeper within the packet [9], [40].

Figure 5 contains an example TCAM table for the central ZG's top interface in Figure 4. Message paths (s, m, r) from the security policy that involve the brake pedal are listed as 128-bit "Allow" entries with an implicit "Deny" for any other message. The first row shows the entry permitting the brake pedal (03:03:03:03:03:03) to send a braking message (ID 347, for example) to the front left wheel (10:10:10:10:10:10).

The TCAM entries enforce check (2), authorization by the security policy. When this TCAM specifying the permitted messages for the brake pedal is applied to the brake pedal interface, it also automatically enforces check (1), authentication. This is because a message from the brake pedal with a different sender would not match any "Allowed" rule in the TCAM. If a switch does not allow interfaces to have separate TCAMs, an automaker could also include an extra "interface ID" byte in the TCAM to achieve the same functionality.

Each switch only needs to implement the relevant portions of the security policy for the ECUs in its zone, which can dramatically reduce the TCAM space required. For example, a ZG with 10 ECUs that each send 10 types of unicast messages would need only 101 TCAM entries ($10 \cdot 10 + 1$ for the implicit deny rule), even if the security policy for the entire vehicle has hundreds more entries.

If there are still too many policy entries to put into the TCAM, automakers can form message ID "families" that have the same upper bits for the switches to treat the same way. For example, if a designer wants to specify 16 different messages that a wheel can send to another ECU, then the designer can use IDs 32–47 for those messages and refer to all of them with a single TCAM field of 0000...00001XXXXX. This lets automakers create flexible rules that save TCAM space.

Although messages only need to be checked at the first ZG they encounter, designers can implement additional TCAM policy checks at the receiving ZGs before forwarding them to the destination ECU. This provides redundancy and allows automakers to create more flexible TCAM schemes.

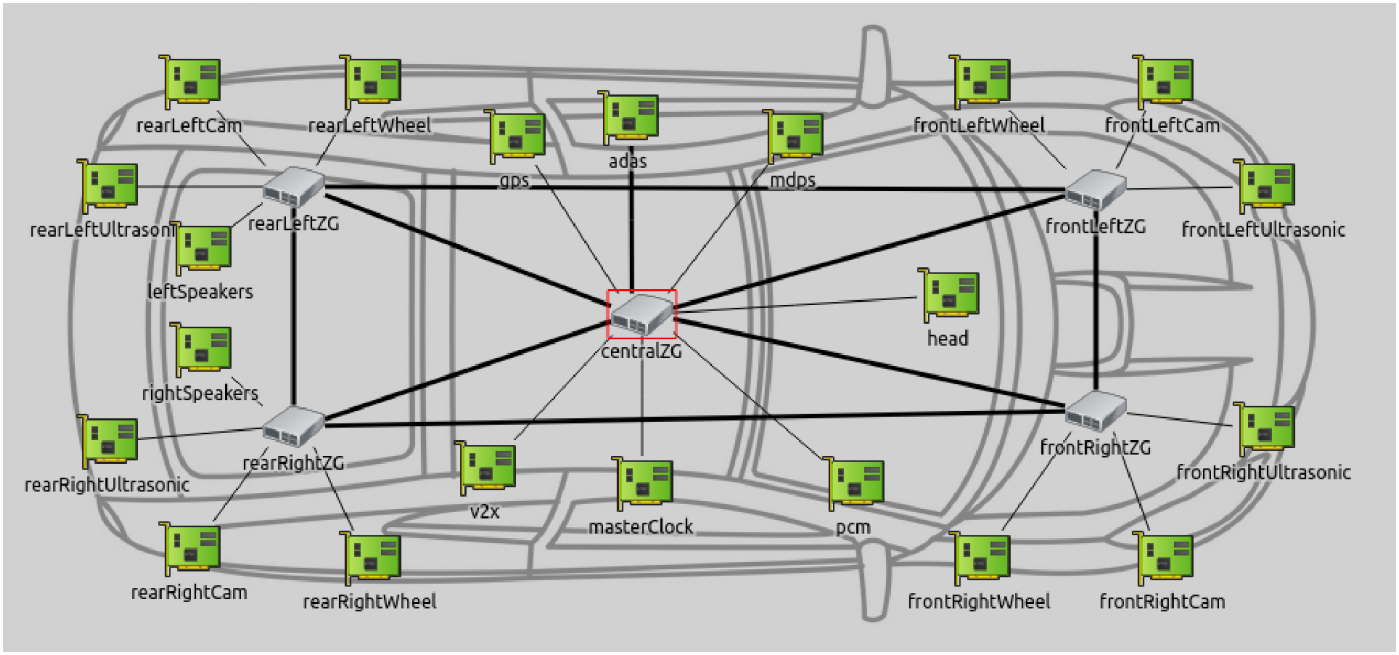


Fig. 6. OMNeT++ model of the IVN. All connections are AE; thin lines are 100BASE-T1 and thick lines are 1000BASE-T1.

TABLE I. NETWORK MODEL TRAFFIC

Stream	Class	Rate (Mbps)	Period (ms)	Size (Bytes)	Deadline (ms)	Path		
Camera Feed	Class B	40	0.25	1250	33	Cameras	→	ADAS
Ultrasonic Feed	Class B	0.256	0.25	8	20	Ultrasonic Sensors	→	Head
Wheel Speed Control	CDT	10	0.5	625	0.1	ADAS	→	Wheels
Engine Control	CDT	0.064	0.5	4	0.1	ADAS	→	PCM
Steering Assist	CDT	10	0.5	625	0.1	ADAS	→	MDPS
GPS Signal	BE	0.00512	100	64	100	GPS	→	Head
V2X Data	CDT	0.256	0.5	16	0.1	V2X	→	ADAS
Audio Data	Class A	0.704	0.125	11	10	Head	→	Speakers

VI. PERFORMANCE EVALUATION

We created a simulation of an AE IVN in OMNeT++ [23] using the INET library [22] to compare the performance overhead of our scheme to alternative security methods. OMNeT++ is a discrete-event simulator, and the INET library provides support for networks, such as the AE IVN we are interested in. Our simulations compare the E2E latency of packets in various streams under both our proposed firewall scheme and a cryptographic authentication scheme based on related work.

A. Network Model

We combined public information from automakers and related simulation research [1], [8], [22], [28] to create a network model that is representative of future zonal IVNs and contains a variety of traffic types from driver assistance, powertrain/chassis, and infotainment domains. Driver assistance ECUs included the advanced driver assistance system (ADAS) supported by cameras, ultrasonic sensors, and a vehicle-to-everything (V2X) antenna. Powertrain/chassis ECUs consisted of wheel speed controllers, the powertrain control module (PCM), and motor-driven power steering (MDPS). Finally, the infotainment ECUs included the head unit, a GPS transceiver, and speakers.

A visualization of our network is shown in Figure 6 with regular traffic flowing between nodes as described in Table I. Each stream in the table is assigned a certain priority class based on its deadline and period: Control Data Traffic (CDT), Class A, Class B, and Best Effort (BE). Class A and Class B traffic refer to the stream reservation classes as defined in the TSN standards [20]. The switches use these classes to prioritize traffic. We connected the ECUs in a combined tree/ring zonal topology for redundancy. All connections used standard 100BASE-T1 except the backbone connections and ADAS connection, which used 1000BASE-T1 due to the amount of data they carried.

B. Firewall Model

Our TCAM-based firewalls are identical to normal switches except for an additional TCAM lookup whenever packets enter and exit the switch. Because TCAMs can complete searches in a single clock cycle, their overhead is constant for each packet. This overhead is generally between 5–100 ns depending on the TCAM architecture [35]. For our model, we chose a conservative estimate of 100 ns.

We simulated the TCAM firewalls by adding a processing delay of 100 ns to all packets (1) entering and (2) leaving a

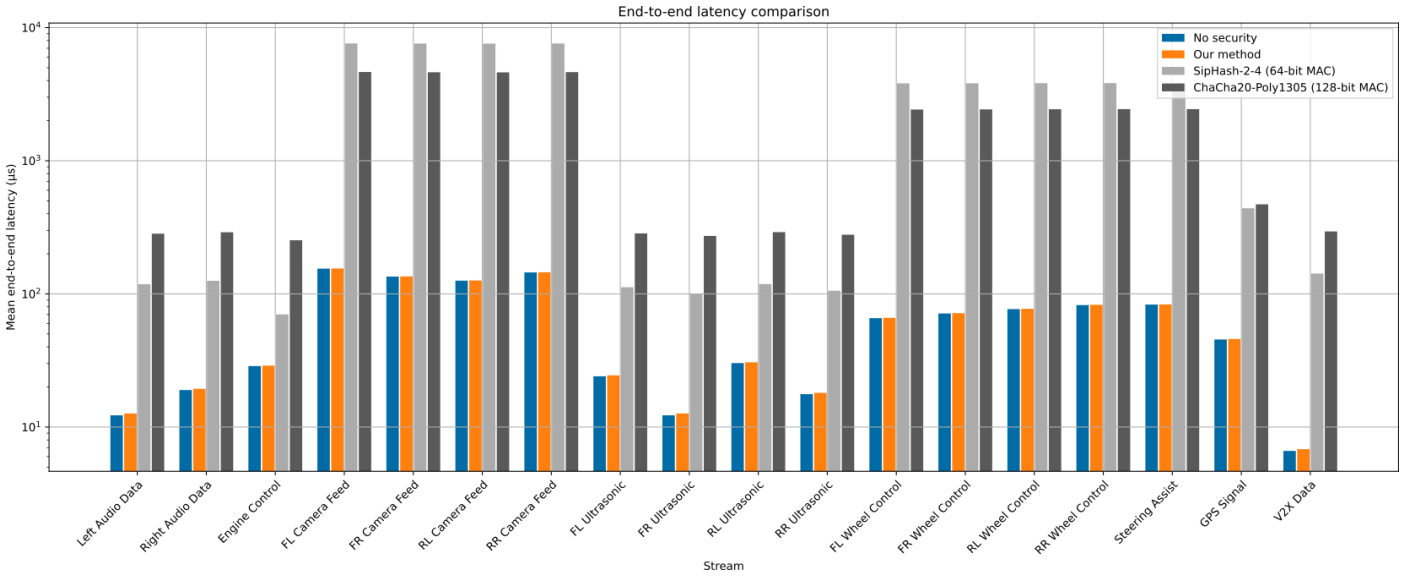


Fig. 7. E2E latency comparison. Each set of bars represents a stream of data from one ECU to another (see Table I). FL/FR/RL/RR mean front left, front right, rear left, and rear right.

ZG. This emulates the "worst-case" performance scenario of an automaker that puts a security TCAM on every interface of every ZG.

C. Cryptography Alternative Model: MAC Authentication

Section IV-A discusses numerous cryptography-based authentication and policy enforcement schemes for IVNs. Nearly all of them authenticate messages using some form of authenticated encryption or MAC generation (HMAC-SHA1, HMAC-SHA256, etc.). The main differences between each of these schemes is usually in how they distribute the symmetric keys and what extra information they include with the transmissions.

To compare our system to these, we construct an "ideal" scenario that represents the fastest possible cryptography-based authentication scheme. Each ECU in this scenario simply appends a MAC to every packet it sends and verifies the MAC on every packet it receives. We assume that somehow the ECUs all share unique, secure symmetric keypairs with each other and that the ECUs do not need to add any extra information to the transmissions. While this may be unrealistic, the intent is to provide a lower bound on the overhead of cryptographic approaches.

To estimate the overhead of generating and then verifying MACs, we use the benchmarks done by Bühler et al. [7] on symmetric cryptography algorithms in embedded settings. Of the different cryptography algorithms they tested, they found that they achieved the lowest latencies on their test microcontroller with SipHash-2-4 [4] (a 64-bit MAC algorithm) and ChaCha20-Poly1305 (which can be used as a 128-bit MAC algorithm). After running a linear regression ($R^2 > 0.999$) on their reported data, we used the latency models in Table II.

We simulated both of these algorithms by adding the appropriate delay as specified in Table II to each sender ECU and receiver ECU for each packet that they process. We also had the sender ECUs append the appropriate number of bytes in the MAC to each packet.

TABLE II. MODELS FOR SELECTED MAC GENERATION ALGORITHMS. x REPRESENTS THE SIZE OF A PACKET PAYLOAD IN BYTES.

Algorithm	Latency Model (µs)	MAC size (bits)
SipHash-2-4	$0.3705x + 20.0$	64
ChaCha20-Poly1305	$0.2125x + 116.4$	128

VII. RESULTS

Figure 7 shows the E2E latency of each stream in the four scenarios we tested: a baseline "no security" case, our TCAM firewall method, and the alternative MAC algorithms SipHash-2-4 and ChaCha20-Poly1305. In every case, our method has a lower E2E latency than the MAC algorithms and nearly matches the latency of the baseline scenario.

In addition, our method meets the deadline requirement for each traffic stream specified in Table I. The only streams for which we measure a latency over 0.1 ms (10^2 µs) are the camera feeds, which have a deadline of 33 ms and likely take longer due to their high packet size (1250 bytes).

The MAC scenarios, however, sometimes create unacceptable latency. For example, both SipHash-2-4 and ChaCha20-Poly1305 exceed the deadline for wheel control by over 10 times. They also are too slow for other CDT streams such as the V2X data, steering assist, and engine control (ChaCha20-Poly1305 only). The MAC algorithms do still satisfy the deadlines of the non-CDT traffic streams.

We focus on the CDT streams more, since they have the most stringent requirements and carry safety-critical traffic. To understand when each method can meet the CDT deadlines of 10^2 µs, Figure 8 shows the E2E latency of the engine control stream as we vary the packet payload size from 4 bytes to 1924 bytes. We retain the same period to simulate control updates that change quickly (every 0.5 ms). We find that our method supports CDT updates of about 825 bytes, matching the base "no security" scenario closely. However, SipHash-

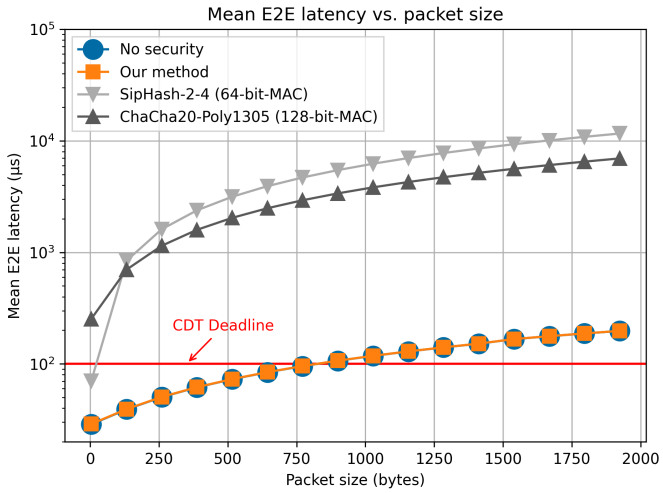


Fig. 8. Mean E2E latency of engine control messages plotted against the size of the messages. The 100µs deadline for CDT traffic is shown.

2-4 can only support CDT updates of about 25 bytes, and ChaCha20-Poly1305 cannot support CDT updates of any size.

VIII. Discussion

The OMNeT++ simulation of the zonal network indicates that our TCAM-based firewall approach allows significantly lower latency traffic to flow through the IVN than cryptography-based protocols. In addition, most actual cryptography-based authentication solutions are not as simple as the MAC generation/verification that we compared against because they must deal with key distribution and freshness, meaning that their overhead may be more than what we measured for the MAC scenarios.

Crucially, our approach allows automakers to *centralize* their security policy enforcement to a set of ZGs that they can more easily control than each edge ECU. Automakers do not have to ensure each of their ECU suppliers adds cryptography support to their constrained microcontrollers. We build our TCAM firewall scheme to be an easy add-on to IVNs with nearly no additional overhead, allowing designers to integrate it with other security protocols if they wish.

A. Limitations and Future Work

Our approach does have the following limitations. First, it does not protect against attacks where a hacker directly compromises a target ECU that is already capable of the accomplishing the goal G that they want. However, we find this scenario to be unlikely, since the most vulnerable targets to a remote attacker are likely to be less-critical ECUs, such as the head unit, that contain wireless transceivers. Other more advanced solutions such as IDPSs may be better at handling this, but they often come with performance concerns.

Second, our approach relies on the ZGs being trusted. If a ZG is compromised, then the network cannot trust that packets from that zone are actually authenticated and our scheme fails. While this scenario is plausible (e.g., in a supply-chain attack), we still find trusting the ZG to be a reasonable and useful assumption. Supply-chain attacks are difficult, especially on a security-essential component such as a ZG. We think it is much more likely that an attacker would instead find a vulnerability in one of the numerous edge ECUs, which our scheme is designed to mitigate. In addition, many current IVN authentication protocols such as Gatekeeper [18] rely on a "trusted" central node like a ZG. Future research could focus on ways to validate the security of a ZG.

Third, we do assume our attacker does not have physical access to the vehicle. This assumption allows us to trust the backbone connections between ZGs. An attacker with physical access could place a node between ZGs and break the scheme. However, we note that this scenario is much less catastrophic than the remote attacks we aim to prevent, such as the Miller and Valasek [26] hack that allowed them to access hundreds of vehicles from across the nation. Physical attacks are much more difficult to prevent—for example, an attacker could just cut the brake lines.

Fourth, our approach requires complete enumeration of the capabilities of each ECU at design time as well as a way to configure the ZGs with the proper rules in the security policy. While this could cause some administrative strain, we argue it is easier than configuring each edge ECU separately with the security policy (which, even if not explicitly defined, always exists implicitly). We posit that enumerating the exact messages each ECU should be able to send and receive will also help automakers make their IVN designs more secure.

Finally, we did not have the resources to build a real IVN and test our firewall approach on actual switches. Future work could create a physical prototype of the system and assess its real-life performance versus simulation.

IX. Conclusions

In this paper, we introduced a scheme to enforce IVN security policies based on a set of distributed TCAM firewalls. Our approach leverages the switched topology of zonal AE networks to authenticate messages from ECUs using the physical interface they are connected to, allowing ZGs to know the true sender of each message without cryptography. We used this functionality to block unauthorized messages from compromised ECUs that do not fit their design, removing a wide variety of options from attackers in the network.

This work used an OMNeT++ simulation of a zonal AE IVN to compare the performance overhead of our approach to more traditional, cryptography-based approaches. We found that our scheme has minimal overhead and is faster than modern lightweight MAC generation/verification algorithms, giving automakers new ways to mitigate attackers while satisfying IVN performance requirements.

Code and Attributions

Code and attributions for icons used to create the figures in this paper can be found at https://github.com/evallen/zonal-filter.

References

[1] Evan Allen, Zeb Bowden, Randy Marchany, and J. Scot Ransbottom. WIP: The Feasibility of High-performance Message Authentication in Automotive Ethernet Networks. In *Proceedings Inaugural International Symposium on Vehicle Security & Privacy*, San Diego, CA, USA, 2023. Internet Society.

[2] Onur Alparslan, Shin'ichi Arakawa, and Masayuki Murata. Next Generation Intra-Vehicle Backbone Network Architectures. In *2021 IEEE 22nd International Conference on High Performance Switching and Routing (HPSR)*, pages 1–7, Paris, France, June 2021. IEEE.

[3] Akib Anwar, Anika Anwar, Lama Moukahal, and Mohammad Zulkernine. Security assessment of in-vehicle communication protocols. *Vehicular Communications*, 44:100639, December 2023.

[4] Jean-Philippe Aumasson and Daniel J. Bernstein. SipHash: A Fast Short-Input PRF. In David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Doug Tygar, Moshe Y. Vardi, Gerhard Weikum, Steven Galbraith, and Mridul Nandi, editors, *Progress in Cryptology - INDOCRYPT 2012*, volume 7668, pages 489–508. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. Series Title: Lecture Notes in Computer Science.

[5] AUTOSAR. SOME/IP Protocol Specification, November 2022. https://www.autosar.org/fileadmin/standards/R22-11/FO/AUTOSAR_PRS_SOMEIPProtocol.pdf.

[6] David Berard and Vincent Dehors. Security of connected vehicles, June 2023. https://www.synacktiv.com/sites/default/files/2023-06/SecuriteDesVoitures.pdf.

[7] Heiko Bühler, Andreas Walz, and Axel Sikora. Benchmarking of Symmetric Cryptographic Algorithms on a Deeply Embedded System. *IFAC-PapersOnLine*, 55(4):266–271, 2022.

[8] Eunmin Choi, Hoseung Song, Suwon Kang, and Ji-Woong Choi. High-Speed, Low-Latency In-Vehicle Network Based on the Bus Topology for Autonomous Vehicles: Automotive Networking and Applications. *IEEE Vehicular Technology Magazine*, 17(1):74–84, March 2022.

[9] Cisco. TCAM Demystified, February 2020. https://learningnetwork.cisco.com/s/article/tcam-demystified.

[10] College of Engineering of Yanbian University, Yanji, 133002, China, Hua Yang, Meng-Zhuo Liu, Yi-Hu Xu, Yu-Jing Wu, and Yi-Nan Xu. Research of Automotive Ethernet Security Based on Encryption and Authentication Method. *International Journal of Computer Theory and Engineering*, 11(1):1–5, 2019.

[11] Division of Electronics and Communication Engineering of Yanbian University, Yanji, China, Jia-Ming Li, Shuo Fu, Yu-Jing Wu, and Yi-Nan Xu. High-Efficiency Encryption and Authentication Network Security for Automotive Ethernet. *International Journal of Modeling and Optimization*, pages 36–43, May 2022.

[12] Aida Ben Chehida Douss, Ryma Abassi, and Damien Sauveron. State-of-the-art survey of in-vehicle protocols and automotive Ethernet security and vulnerabilities. *Mathematical Biosciences and Engineering*, 20(9):17057–17095, 2023.

[13] Morris J Dworkin. Advanced Encryption Standard (AES). Technical Report NIST FIPS 197-upd1, National Institute of Standards and Technology, Gaithersburg, MD, 2023.

[14] Alessandro Frigerio, Bart Vermeulen, and Kees G. W. Goossens. Automotive Architecture Topologies: Analysis for Safety-Critical Autonomous Vehicle Applications. *IEEE Access*, 9:62837–62846, 2021.

[15] Andy Greenberg. GM Took 5 Years to Fix a Full-Takeover Hack in Millions of OnStar Cars, September 2015. https://www.wired.com/2015/09/gm-took-5-years-fix-full-takeover-hack-millions-onstar-cars/.

[16] Andy Greenberg. The Jeep Hackers Are Back to Prove Car Hacking Can Get Much Worse, August 2016. https://www.wired.com/2016/08/jeep-hackers-return-high-speed-steering-acceleration-hacks/.

[17] Jan Holle and Siddharth Shukla. Gatekeeper for In-vehicle Network Communication. *ATZelektronik worldwide*, 13(6):40–43, December 2018.

[18] Shengtuo Hu, Qingzhao Zhang, André Weimerskirch, and Z. Morley Mao. Gatekeeper: A Gateway-based Broadcast Authentication Protocol for the In-Vehicle Ethernet. In *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '22, pages 494–507, New York, NY, USA, May 2022. Association for Computing Machinery.

[19] Martin Husák, Giovanni Apruzzese, Shanchieh Jay Yang, and Gordon Werner. Towards an Efficient Detection of Pivoting Activity. In *2021 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pages 980–985, 2021.

[20] IEEE. *802.1Q-2022 - IEEE Standard for Local and Metropolitan Area Networks–Bridges and Bridged Networks*. IEEE, 2022. OCLC: 1396933774.

[21] IEEE. *802.3-2022 - IEEE Standard for Ethernet*. Number 802.3-2022. IEEE, 2022. OCLC: 1343075176.

[22] OpenSim Ltd. INET Framework, 2023. https://inet.omnetpp.org/.

[23] OpenSim Ltd. OMNeT++, 2023. https://omnetpp.org/.

[24] Feng Luo and Shuo Hou. Security Mechanisms Design of Automotive Gateway Firewall. pages 2019–01–0481, April 2019.

[25] Bin Ma, Shichun Yang, Zheng Zuo, Bosong Zou, Yaoguang Cao, Xiaoyu Yan, Sida Zhou, and Jichong Li. An Authentication and Secure Communication Scheme for In-Vehicle Networks Based on SOME/IP. *Sensors*, 22(2):647, January 2022.

[26] Charlie Miller and Chris Valasek. Remote Exploitation of an Unaltered Passenger Vehicle, August 2015.

[27] Y. Nir and A. Langley. ChaCha20 and Poly1305 for IETF Protocols. Technical Report RFC7539, RFC Editor, May 2015.

[28] Chulsun Park and Sungkwon Park. Performance Evaluation of Zone-Based In-Vehicle Network Architecture for Autonomous Vehicles. *Sensors*, 23(2):669, January 2023.

[29] Mert D. Pesé, Karsten Schmidt, and Harald Zweck. Hardware/Software Co-Design of an Automotive Embedded Firewall. pages 2017–01–1659, March 2017.

[30] Dominik Reinhardt, Maximilian Guntner, Markus Kucera, Thomas Waas, and Winfried Kuhnhauser. Mapping CAN-to-ethernet communication channels within virtualized embedded environments. In *10th IEEE International Symposium on Industrial Embedded Systems (SIES)*, pages 1–10, Siegen, Germany, June 2015. IEEE.

[31] Dominik Reinhardt and Markus Kucera. Domain Controlled Architecture - A New Approach for Large Scale Software Integrated Automotive Systems:. In *Proceedings of the 3rd International Conference on Pervasive Embedded Computing and Communication Systems*, pages 221–226, Barcelona, Spain, 2013. SciTePress - Science and and Technology Publications.

[32] Erik Shilling. It Took Under Two Minutes For THese Hackers to Hack a Tesla Model 3, March 2023. https://jalopnik.com/it-took-under-two-minutes-for-these-hackers-to-hack-a-t-1850275174.

[33] Edilson A. Silva, Paulo Freitas De Araujo-Filho, and Divanilson R. Campelo. Experimental Evaluation of Cryptography Overhead in Automotive Safety-Critical Communication. In *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*, pages 1–5, Porto, June 2018. IEEE.

[34] Neha Nikesh Surjekar, Yutika Patwardhan, and Vinaykumar Konduju. A CASE STUDY ON MIGRATING TOWARDS FUNCTIONALLY SAFE ZONAL ARCHITECTURE USING MBSE. *INCOSE International Symposium*, 33(1):1403–1417, July 2023.

[35] Inayat Ullah, Zahid Ullah, Umar Afzaal, and Jeong-A Lee. DURE: An Energy- and Resource-Efficient TCAM Architecture for FPGAs With Dynamic Updates. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 27(6):1298–1307, June 2019.

[36] Vector. CANoe.Ethernet. https://www.vector.com/us/en/products/products-a-z/software/canoe/option-ethernet/#c91477.

[37] Chu-Ting Wang, Gui-He Qin, Rui Zhao, and Shi-Min Song. An Information Security Protocol for Automotive Ethernet. *Journal of Computers*, 32(1):39–52, 2021.

[38] Eric Wang, William Xu, Suhas Sastry, Songsong Liu, and Kai Zeng. Hardware module-based message authentication in intra-vehicle networks. In *Proceedings of the 8th International Conference on Cyber-Physical Systems*, pages 207–216, Pittsburgh Pennsylvania, April 2017. ACM.

[39] Elizabeth Weise. Chinese group hacks a Tesla for the second year in a row, July 2017. https://www.usatoday.com/story/tech/2017/07/28/chinese-group-hacks-tesla-second-year-row/518430001/.

[40] Eren Yilmaz. *Firewall and Intrusion Detection and Prevention Concept for Automotive Ethernet*. PhD Thesis, 2020. Series: IT.