Advanced IDPS Architecture for Connected and Autonomous Vehicles

Sherin Kalli Valappil*, Lars Vogel*, Mohammad Hamad†, and Sebastian Steinhorst†
*Robert Bosch GmbH, Abstatt, Germany
†Technical University of Munich, Munich, Germany

***Abstract*—Highly connected and automated driving technologies have ushered digital transformation and flexibility to modern cars. However, the vehicle's attack surface has significantly expanded due to increased connectivity. To address this problem, automotive manufacturers are adopting more secure practices driven by standards and regulations. In addition to the deployed cryptographically strong security measures in automotive, we need an Intrusion Detection and Prevention System (IDPS) that actively monitors the vehicle for intrusions, prevents them, and provides notification, as required by UN Regulation No. 155. In this work, we aim to identify the current limitations of the existing automotive approaches and contribute to an advanced IDPS solution. We propose architectural changes that improve reliability and form a framework to propose reactions in a safety-related automotive context. We evaluate our proposed architecture with regard to performance and security design. With the proposed changes to the IDPS architecture, our aim is to integrate a dynamic and adaptive strategy for IDPS, enhancing resilience against emerging threats and vulnerabilities.**

***Index Terms*—Security, Intrusion Detection and Prevention, Connected and Autonomous Vehicles**

I. Introduction

Advanced technologies in today's highly connected and autonomous vehicles have increased their susceptibility to numerous cyberattacks. Detecting intrusions and responding to these cyber threats has become both critical and challenging [1]. In response to this concern, the recent UN Regulation 155 [2] mandates manufacturers to establish strong cybersecurity management systems, conduct risk assessments, and enforce stringent security measures.

The AUTOSAR Intrusion Detection System Manager (IdsM) [3], a standard introduced by AUTOSAR, provides a systematic approach to standardizing the IDPS for automotive applications on classic platforms. While the AUTOSAR IdsM effectively manages the buffering and filtering of security events reported by software applications, it does not have the capability to definitively confirm the presence of an intruder. Moreover, the existing AUTOSAR architecture does not provide isolation between the IdsM and other applications hosted on the same Electronic Control Unit (ECU). As a result, the integrity of the co-hosted IDPS could be at risk if any of these applications are compromised. In safety-critical environments like connected and autonomous vehicles, the brief time available to respond to an attack and secure the system highlights the need for reliably identifying an intruder's presence within the ECU. This requires comprehensive offline analysis and the ability to respond to intrusions within the limited time frame. Neglecting these challenges may have severe consequences, jeopardizing vehicle safety.

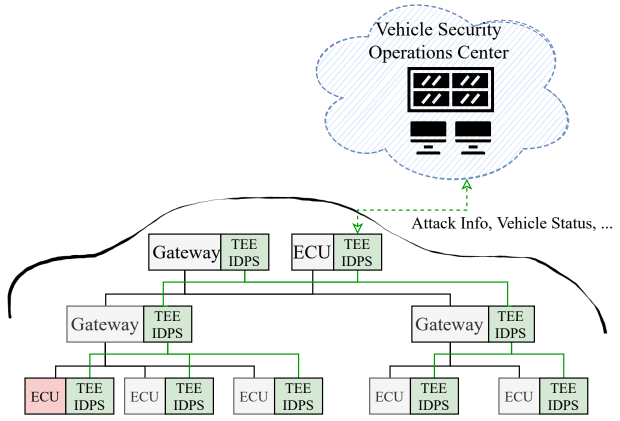

Fig. 1: High-level representation of our proposed IDPS within the vehicle E/E architecture, highlighting isolated communication channels between different IDPSs and VSOCs. Even if one ECU is compromised (in red), its IDPS remains secure, ensuring secure communication of its status.

To tackle these challenges, our paper explores strategies to elevate the standard of automotive IDPS solutions. Firstly, we propose an architecture, depicted in Fig. 1, designed to uphold IDPS reliability, even in the event of an ECU compromise. Secondly, we present an approach to enhance attack detection and confirmation by analyzing patterns derived from reported security events. To this end, we suggest tailoring MITRE frameworks for a comprehensive threat model that surpasses the constraints of the traditional Threat Assessment and Risk Analysis (TARA) approach. While TARA addresses external threats, our proposed approach broadens the scope to include internal threats. Additionally, our solution's dynamic nature addresses Advanced Persistent Threat (APT), contrasting the static analysis provided by conventional automotive threat models. This dynamic methodology enhances the capability to analyze evolving threats in the automotive domain, thereby fortifying cyber resilience. In summary, in this paper:

- We propose an architecture for automotive IDPS that reflects the advanced threats faced in the dynamic automotive landscape (Sec. III and IV).
- We implement our architecture, empirically assess detection and mitigation approaches, and propose enhance-

ments to enhance security, emphasizing key decisions. (Sec. V).

II. System and Attacker Model

System Model: The automotive Electric and Electronic (E/E) architecture consists of numerous ECUs interconnected through diverse networks like CAN, Ethernet, and FlexRay. These ECUs vary in safety integrity levels. With some of the ECUs adhere to AUTOSAR standards for safety-critical functions, while others, particularly microprocessor-based ECUs utilize POSIX-based Operating Systems (OSs). This work focuses on microprocessor-based ECUs known for their complex implementation and vulnerabilities as potential attack points due to their connectivity to the external environment. Despite the diversity in OSs, our work proposes a standardized communication protocol aligned with the frame format of the IdsM protocol as defined by AUTOSAR [4]. In our system model, a designated ECU establishes the communication with the Vehicle Security Operations Center (VSOC), facilitating data exchange within the automotive E/E architecture [5].

Attacker Model: This section provides an overview of our considered attacker model. We assume that each ECU has state-of-the-art security mechanisms in place, including Secure Boot, Message authentication for communication via Secure Onboard Communication (SecOC) [6], software signature verification before flashing, secure access to memory and diagnosis, sensitive and key material protection, and usage of Hardware Security Module (HSM) for tamper-resistant storage and cryptographic computations [7].

While our security measures are in place, we acknowledge the constant threat of attackers seeking to exploit implementation weaknesses and vulnerabilities. These vulnerabilities could lead to safety-critical manipulations, impacting components like brakes, steering, and acceleration, as evidenced by Miller et al. [8]. Attackers can also exploit ECUs for lateral movements and pivoting attacks, navigating within an ECU and extending to other ECUs. This was demonstrated by the Tesla Free-Fall attack [9], which compromised one or more ECUs, leading to a loss of control and safety. The threat landscape also includes malicious activities, such as leveraging control of the ECU to extort sensitive and confidential data for ransomware attacks [10]. Given this attacker's intent, our model assumes the potential compromise of the entire ECU. Consequently, if the ECU is compromised, the communication channel and memory are also compromised. We assume the integrity of communication with the back-end is protected. However, an attacker could obstruct the ECU from sending information to the VSOC, preventing timely responses from the VSOC. We exclude sophisticated attacks, like man-in-the-middle attacks, aiming to intercept communication sent to the VSOC and other ECUs. Additionally, we assume the use of a Trusted Execution Environment (TEE) and trust that the TEE remains uncompromised. We posit that an attacker compromising the host application cannot breach the trust boundary.

III. Preliminaries

In order to mitigate attacks through an IDPS in an automotive setting, selecting a threat model aligned with potential intrusions is crucial. A systematic analysis of the chosen threat model allows for identifying suitable detection methods and proposing effective countermeasures against potential threats. Currently, threat modeling in the automotive industry primarily relies on TARA. However, TARA mainly focuses on external threats, resulting in a limited threat landscape reflected in the IDPS's capabilities. Therefore, a comprehensive offline analysis, supported by an appropriate framework and knowledge base, is necessary when designing an IDPS for automotive systems. To achieve this, we have explored Cyber Kill Chain (CKC), MITRE Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK), and D3FEND models.

The CKC, introduced in 2011 [11] and utilized in Cyber Threat Intelligence (CTI), outlines that an attack follows a sequential chain or end-to-end process, involving seven stages: *weaponization, delivery, exploitation, installation, command and control (C2), and actions on objectives.* Analyzing these stages within the CKC allows us to predict the current stage of an attacker's progress. Understanding these adversary steps in detail enables the introduction of security decisions and measures to identify Indicators of Compromise (IoC).

The MITRE ATT&CK framework offers a comprehensive examination of the attack lifecycle based on the attacker's tactics to identify IoCs. Compared to CKC, the MITRE framework provides a more detailed breakdown of attack stages, offering finer granularity that is beneficial for IDPS threat modeling and covering a broader threat landscape. MITRE ATT&CK is a behavior-based approach, categorizing tactics as the 'why' (reason an attacker performs a specific action in threat modeling) and techniques as the 'how' (method by which an action was performed). By combining these techniques and tactics with the target, we can deduce *"what"*-was performed on the target [12]. Inspired by MITRE ATT&CK, a domain for automotive systems is under development by Auto-ISAC, as mentioned by Mackay [13]. The stages in the MITRE framework for automotive systems include: *Manipulate Environment, Initial Access, Execution, Persistence, Privilege Escalation, Defense Evasion, Credential Access, Discovery, Lateral Movement, Collection, Command and Control, Exfiltration, Affect Vehicle Function, and Impact.* MITRE also offers Detection, Denial, and Disruption Framework Empowering Network Defense (D3FEND) [14], functioning as a mitigation matrix database or knowledge graph. This tool provides cybersecurity countermeasures to help mitigate detected attacks.

We analyze automotive MITRE ATT&CK techniques along with their corresponding entries in D3FEND to support the design of the IDPS. This involves examining various security attacks that have targeted automotive systems [8], [9], [15], [16], including a notable example - the Tesla Free-Fall WiFi attack [9]. This real-world incident emphasizes the need for in-depth detection and mitigation strategies in our proposed IDPS

TABLE I: Tesla Free-Fall attack visualized with MITRE ATT&CK techniques and mitigation identified through MITRE D3FEND.

Attack Stages	**Technique**	**Detection and Mitigation**
Vulnerability scanning	Network traffic analysis	Configuration hardening
Rougue WiFi Access Point (Tesla Guest)	Rogue WiFi Access Point	Configuration hardening
Buffer overflow - code injection on `QtCarBrowser` (CVE-2011-3928)	Exploit public facing application	Application hardening through stack protections
ARM vulnerability on Linux (CVE-2013-6282) allowing `UID: 2222` to become root through `put_user/get_user`	Abuse Elevation control mechanism	Isolation through system call filtering
Disables `AppArmour`: `reset_security_ops()`	Bypass mandatory access control	Access control and isolation via system call filtering
ECU verifies only filename before flashing	Bypass code signing	Implementation of cryptographic code signing
Update Gateway ECU `gtw.hex` rules	Reprogram for lateral movement	Implement secure flashing approach to authorize
Manipulate brakes and steering on CAN	Modify Bus message	Share the compromised information from GW to other ECUs
Free-Fall of messages from WiFi to CAN	Loss of control / safety	Retain safety

design. The attack unfolded in progressive stages, starting with vulnerability analysis and culminating in the complete compromise of vehicle control or safety.

Table I shows the attack stages carried out by the attackers, mapped to MITRE ATT&CK and D3FEND. From the information in column 2 of Table I, it becomes evident that a specific ECU may face numerous threats, systematically identified through an in-depth threat modeling approach implemented using MITRE ATT&CK. Once all the threats and exploitation techniques are identified, corresponding detection and mitigation strategies can be developed, leveraging the D3FEND framework, as shown in column 3 of Table I. Additionally, the example in the table highlights its capability to address threats involving lateral movements. Thus, by correlating the attack technique with the attack stage, insights into the attacker's intent and the severity of the attack can be deduced.

The detection and mitigation techniques we have selected form the foundation of our proposed IDPS. These selections are based on the frequency and severity of attacks in the automotive sector. The techniques include *cryptographic measures, application hardening, process isolation, and network analysis from the host's perspective*. Other techniques, such as message authentication, are already part of the state-of-the-art in automotive security. In the following section, we introduce the terminologies used in IDPS, describe our intrusion detection and prevention elements, outline our selected detection and mitigation strategies, and illustrate how they are integrated into our automotive architecture.

IV. Proposed Architecture

The proposed architecture is shown in Fig. 2, depicting a microprocessor-based environment commonly found in complex automotive ECUs. This environment is capable of hosting multiple virtualized instances, such as an AUTOSAR Adaptive POSIX-based OS for safety-related applications and a Linux-based system for AI/ML applications. Our approach integrates a TEE alongside a hardware cryptographic co-processor housed in a HSM and multiple Virtual Machines (VMs) within a System on a Chip (SoC) framework. This architecture enables the secure integration of intrusion detection elements. The TEE, denoted as *IDM TEE* in our architecture, incorporates the detection engines, internal forensics, and reporting module responsible for back-end communication. Crucially, the communication context of the TEE is isolated from the Rich Execution Environment (REE). Therefore, even if the host environment is compromised, communication to the back-end remains secure and uncompromised. The internal elements facilitating the implementation of intrusion detection and ensuring overall security are detailed in the subsections below.

Security Policy: To achieve application hardening, implementing an access control mechanism becomes imperative. This mechanism enforces a Security Policy (secpol) dictating the permissible activities within the application [17]. We implement access control mechanisms using existing secpol, such as AppArmor in Linux and security policy in QNX. The secpol defines rules and access rights for various system components. The kernel manages, securely stores, and verifies the secpol during system boot-up. Post the startup of the OS, security policies are loaded, and rights for individual objects are retrieved. Only the Policy Engine in the kernel can read these policies, and log files are exclusively writable by the policy server, preventing compromise and log file corruption by a compromised user.

Execution Manager: The Execution Manager, modeled after AUTOSAR Adaptive Platform's Execution Manager, is the initial executable launched post-OS. It manages ECU applications, addressing configuration properties, resource group allocation, and startup/shutdown procedures. Utilizing a manifest, the Execution Manager governs process-specific attributes such as priority, scheduling policy, and access rights [18]. This design ensures fine-grained access control over the application and its resources.

Strict measures disallowing external spawning of processes, as implemented in our system, represent a novel addition to the established AUTOSAR framework. This enhancement effectively mitigates security threats, including process and binary injections, through the implementation of robust access controls.

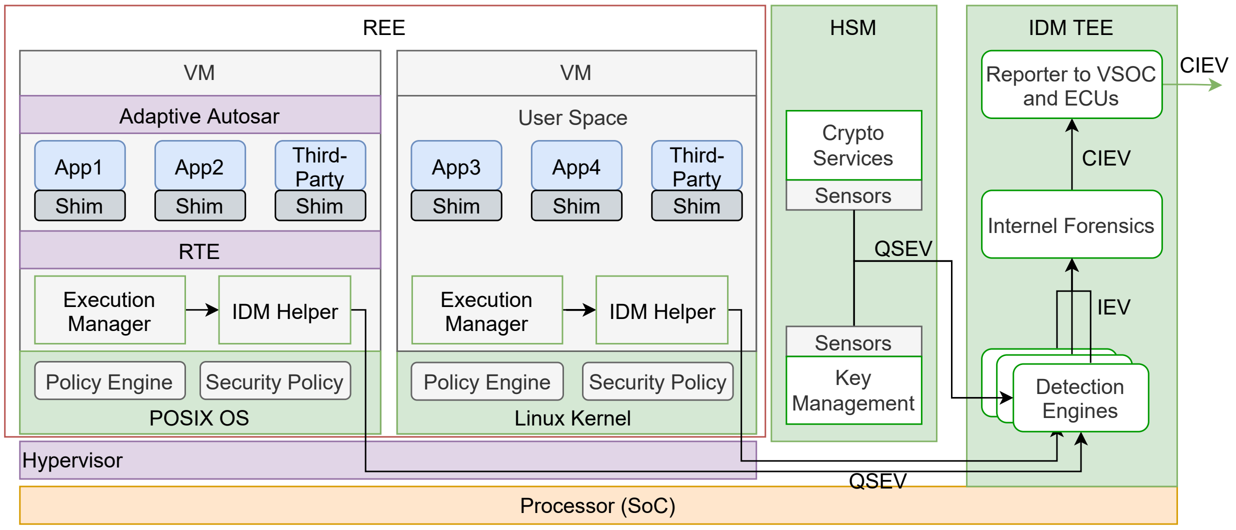

Fig. 2: Proposed IDPS architecture showing the internal components represented on a virtualized SoC platform.

***Sensors and Shims**:* Additional code integrated into the binary for intrusion detection is known as shims [19]. The effectiveness of intrusion detection heavily depends on a comprehensive analysis of the need for these shims. Without them, the intrusion detection system may not reliably detect specific cyberattacks. To create the sensors and shims, we analyzed automotive attacks and experimental studies [8], [9], [15], [16]. We developed our sensors and shims using the MITRE D3FEND framework, which facilitates development of sensors through measures such as *hardening, detection, isolation, eviction, deception, and restoration* techniques. Sensors and shims are categorized based on their deployment location, either as network deployment sensors or host deployment agents, according to NIST [19].

1) Network Sensors: The network serves as the initial point of contact with the outside world, making ECUs connected to the external world, such as Infotainment and Telematics, vulnerable to attacks. To strengthen these complex ECUs, network-based IDPS are developed in conjunction with firewalls [20], [21]. The network data of in-vehicle automotive architectures is defined in files such as AUTOSAR XML (arxml) or database CAN (DBC). These files form the basis for the development of network sensors through the analysis of message properties like frequency and data length, defined in the network matrix. This information can be used to develop statistical anomaly detection, whitelisting, frequency analysis, and analysis of diagnostic requests and their Negative Response Codes (NRC) [22]. Through Deep Packet Inspection (DPI), the network data is analyzed, and sanitization techniques are deployed. This paper does not delve into the network-based sensors in depth; instead, it assumes the availability of certain sensors that are commercially available [20].

2) Host Agents/Shims: AUTOSAR IdsM enables event reporting at the software component level, capturing events that include software-related errors. Therefore, our focus is on customizing agents at the software level to gather information relevant to potential attacks. Utilizing the MITRE ATT&CK and D3FEND approach, we delved into buffer overflow mitigation and process isolation techniques through security policies, referencing the work by [8], [9], [15], [16], [23]. Among the application hardening techniques, we investigated stack protection through binary instrumentation provided by the GCC compiler option. Our experiments included compiler options like *-fstack-protector-all* for Stack Smashing Protector (SSP), `-fPIE` and `-fPIC` for Address Space Layout Randomization (ASLR), and `-fcf-protection=full -flto -O1` for Control-Flow Integrity (CFI). For process isolation, we deployed a Discretionary Access Control (DAC)-based access control using Linux user rights and permissions. Binaries launched by the execution manager are provided with limited access rights to resources and files. D3FEND elaborates on several other mitigation techniques that can be deployed depending on the type of threat that the target ECUs face.

3) Cryptographic Sensors: The cryptographic events reported by cryptographic co-processors, such as the HSM, are of high security importance. State-of-the-art cryptographic measures like SecOC for message authentication [6], software signing, and authenticated security access [22] are adopted for automotive systems [7]. These measures are used for diagnosis, secure flashing, and secure software updates. The combination of these security events, along with other network and host events, provides valuable insight into potential intrusions.

***IDM Helper**:* The IDM Helper is a specialized component designed to limit communication with the TEE. Events detected by the Execution Manager are reported to the IDM Helper. In addition, the IDM Helper periodically monitors the log at specified intervals to detect any potential intrusions. Upon receiving active sensor information, the IDM Helper assesses the event's criticality based on a predefined list. This information undergoes immediate validation by analyzing the logs, and the IDM Helper then forwards this validated information to the IDM TEE. The communication to the IDM TEE is isolated by hardware properties, such as the TrustZone Secure Monitor Calls. This ensures the secure transmission of information to IDM TEE.

Detection Engines: Detection engines play a crucial role in analyzing collected events to identify potential intrusions. The use of multiple detection engines enhances the quality and accuracy of intrusion detection. Various types of detection engines, such as anomaly detection, signature or pattern detection, and rule-based detection engines, can be employed. Due to the detection accuracy and complexity of anomaly-based approaches, we have chosen to focus on pattern and rule-based intrusion detection for this work. The detection engines combine multiple security events to determine the intrusion attempt. In AUTOSAR IdsM, some events are buffered, filtered, and immediately qualified as qualified security event (QSEV). In our proposed system, these QSEVs are further processed by the detection engines to match with patterns and rules, forming Intrusion Events (IEV).

Internal Forensics: Internal forensics enables attack analysis, evaluating each detected intrusion's severity before the IDM unit suggests a reaction. The proposed reaction depends on the safety level of the underlying ECU. Internal forensics, being a future topic, is expected to enhance the prevention capabilities of intrusion detection and prevention systems by providing timely responses. The events reported by the internal forensics are classified as Confirmed Intrusion Event (CIEV).

Intrusion Reporter: The intrusion reporter component facilitates information sharing among the various IDPS in different ECUs and to the back-end, the VSOC. The exchanged data adheres to the same format as the Intrusion Detection System Protocol of AUTOSAR [4], ensuring a standard communication between different ECUs. The frame's integrity and freshness are protected, and it can be encrypted for secure transmission over a wireless network. A crucial security aspect of the Intrusion Reporter in our architecture is that the reported information is isolated from the host system, ensuring a trusted architecture. This provides a medium for reporting the intrusion status between ECUs and VSOC. Upon receiving the intrusion status, each ECU can decide on the reliability of data received from a potentially compromised ECU and trigger degradation modes if necessary.

V. Evaluation

This section details the important aspects of the architecture we evaluated within this work.

IDPS Prototype: A prototype of our proposed solution has been developed within a virtualized Linux environment, designed to closely resemble the complexity of an automotive ECU setup. However, the environment does not fully replicate an ECU system, preventing the thorough testing of the entire Tesla free-fall attack discussed before. During this phase of our research, our primary focus has been on the implementation and evaluation of key components of the proposed system, specifically the shims, security policy and engine and the execution manager. This approach allows us to ensure the robustness and effectiveness of each element before integrating them into a comprehensive solution. As we progress, we anticipate expanding our scope to encompass other components, thereby enhancing the overall functionality and resilience of our system.

Detection Evaluation: In our study, we assume that proficient attacker utilize various tools, such as fuzzing and Common Vulnerabilities and Exposures (CVE) exploitation, to identify system vulnerabilities. To assess the effectiveness of our detection system against these threats, we have devised test cases targeting intentionally introduced vulnerabilities. For this, we deliberately inserted buffer overflow vulnerabilities into the sample application to simulate potential attack scenarios. The sample application, representing a typical automotive software environment, serves as the foundation of our evaluation. Within this application stack, we simulate a communication stack with packet encryption and decryption, reflecting real-world scenarios.

Furthermore, we implemented a series of protective measures, known as shims. These shims, including stack canaries (SSP), ASLR, and CFI, function as proactive agents designed to detect and mitigate heap and stack vulnerabilities within our automotive setup, which are among the most common vulnerabilities in this domain [24]. The binaries associated with these vulnerabilities were compiled using GNU Compiler Collection (GCC), a widely adopted standard in the development of ECUs within the automotive domain.

We evaluated the detection efficiency of the proposed system by first attempting to spawn a process that had not been previously defined. Our observations revealed that the system successfully prevented this unauthorised spawning. In addition, we tested the access control mechanisms implemented through the security policy by initiating processes with resources that had not been previously allocated and attempted to grant additional permissions. During run-time, we also tried accessing files and Interprocess Communication (IPC) channels that were not permitted in the access control list, which led to observable error numbers returned to the execution manager, as expected. With respect to buffer overflows, we manipulated the packets which the application receives, with out-of-range data, and found that these were effectively mitigated by the implemented buffer overflow protections via the shim, leading to Standard Errors (STDERRs) that can be further processed and reported by the IDPS reporting system.

Performance Evaluation: The complete system's performance evaluation was not feasible in this stage of our research. Instead, we explored methods to generate figures closely representing automotive use cases. For this purpose, we examined an Automotive real-time ECU implementing a Driver Assistance System, operating on a QNX system (project details are kept confidential). The QNX system is equipped with a pre-existing default security policy known as *secpol*. This policy effectively manages access to objects and resources by defining rulesets. To assess the differences

TABLE II: Performance evaluation of shims instrumentation against the reference value of *None*, indicating the absence of instrumentation.

Shim	**Text (Bytes)**	**Data (Bytes)**	**Cycle Count**	**Runtime Overhead (%)**
None	4947	712	2146	0.0
SSP	4963	712	2356	8.9
ASLR	5262	768	2738	27.5
CFI	5075	712	2485	15.7

between the default and comprehensive QNX security policies, we recorded tracebuffers during the boot procedure until a specific process successfully started. This allowed us to analyze and compare the implementations of these security policies and their impact on time consumption. During our analysis using the Tracelogger tool, we observed minimal differences in performance values when specific *secpol* was enabled or disabled. This observation provided confidence that our conceptual implementation could be executed without significant additional overhead.

Subsequently, to evaluate the potential performance overhead introduced by the shim, we tested the buffer overflow mitigations under different options, specifically ASLR, SSP, and CFI, on software compiled with GCC. Table II presents the evaluation results, detailing the overhead incurred by the mitigation methods in terms of additional code added to the binaries, increased code size, and additional run-time overhead, compared to a non-existent instrumentation referred to as ”None” in the table. The evaluation shows that the use of ASLR results in the highest run-time overhead of 27.5%, followed by CFI and SSP with overheads of 15.7% and 8.9%, respectively.

Security Design Evaluation: Ensuring the robustness of our proposed concept was a primary objective in our architectural design. Therefore, we conducted an evaluation using the STRIDE-per-interaction methodology, facilitated by the Microsoft Threat Modeling tool [25]. This assessment allowed us to define trust boundaries and make crucial design decisions. The following enumeration outlines the key design decisions that resulted from our evaluation process.

- Hosting intrusion detection components in an isolated TEE environment is imperative for robust security.
- The Execution Manager should exclusively maintain the list of processes to be launched, thereby preventing other applications from initiating a process.
- It’s essential to extend access control for resources to all applications, including third-party applications.
- Ensuring access rights for security policy storage is crucial. Even modifications by root users should be restricted.
- The security policy should be prioritized to be loaded immediately after boot-up, preceding any application loading.
- Modifications to access logs should only be permissible through the security policy enforcement server.

These design decisions significantly enhance the architecture’s robustness by reinforcing critical security properties.

VI. Related Work

The NIST guide on IDPS [19] serves as a vital reference for automotive IDPS in IT systems. The initial insight into the application of IDPS to automotive systems was first analyzed by Hoppe et al. [26] in 2009. Their paper highlighted the need for reactive measures through IDPS, complementing the existing proactive measures. Lee et al. [27] evaluated the necessity of IDPS in addressing automotive attacks. They utilized the cyber kill chain from Lockheed Martin Cooperation [11] and derived a strategic course of action for vehicle security.

Several network-based IDPS solutions, such as Snort [28] and CycurIDS [20], have been extensively researched for both IT and automotive applications. While a network-based IDPS focuses on external threats, it is crucial to complement it with a host-IDPS to provide an effective defense strategy. Wasicek et al. [29] developed an Artificial Neural Network (ANN)-based intrusion detection system, placing the IDS in the central ECU and modeling vital vehicle parameters to detect chip tuning. Jiang et al. [30] expanded this by incorporating road context, using camera images in a Convolutional Neural Network (CNN) model for intrusion identification. The explainability of machine learning-based intrusion detection and the challenges of false positive rates in anomaly-based classification, as noted by [31], make deploying preventive measures challenging. The use of a distributed intrusion detection system, hosted by each ECU, was proposed in [21]. However, this solution was limited to monitoring the network traffic of each ECU and did not consider other types of attacks.

The use of an isolated, tamper-resistant core for the IDPS was proposed by Yoon et al. [6]. A similar architecture is found in ETAS CycurSoC for automotive applications [32], which includes TEE and REE partitions along with HSM. While the NIST’s IDPS guideline [19] provides valuable insights on architectural design, the same architecture may not be suitable for scenarios involving a compromised host. Hoppe et al. [26] examined the integration of diverse detection engines. We propose to expand these detection engines to address automotive threats with the assistance of MITRE ATT&CK and D3FEND. For the forensics-based reaction approach proposed by Hamad et al. [1], our proposed architecture forms a robust framework through isolation mechanisms.

VII. Conclusion and Future Work

In this work, we analyzed the critical aspects of automotive IDPS and the improvements in the current AUTOSAR standard for intrusion detection. We proposed using an in-depth threat analysis approach, such as MITRE, in conjunction with TARA to address both insider and outsider threats, thereby enhancing the resilience of ECUs. We proposed an architecture for IDPS that enables communication of the health status of the ECUs with other ECUs and to VSOC, even if the host of the ECU is compromised. This allows the VSOC to propose a reaction strategy, and the other ECUs to respond accordingly with the intrusion status received from the compromised ECU. We evaluated some proposed mitigation strategies in a virtualized environment and observed that employing sensors introduces additional overhead. We additionally evaluated the proposed architecture using STRIDE-per-interaction to identify crucial design decisions to enhance resilience of our architecture.

The performance results show that while the mitigation methods proposed can enhance security, they come with a performance cost. The overhead needs further evaluation for the corresponding ECU, taking into account anticipated threats and the available computational capacity. Such a prevention technique might not be implemented on a classic AUTOSAR-based ECU running on a micro-controller due to its computational limitations. Simultaneously, the threats faced by such an ECU, compared to a complex ECU with external connections, are minimal, allowing for an evaluation of associated risks. As future work, we intend to evaluate the concept on a complex real target ECU, such as an infotainment system, to analyze the threats faced and overall overhead observed as well as the reaction safety.

Furthermore, the forensic unit represents a crucial next step in our work. Depending on the state of the vehicle and the stage of the attack, the severity of the intrusion may vary. By integrating it with the safety context of the vehicle, a reaction can be proposed within the vehicle, addressing the available time window. This facilitates an internal reaction strategy, eliminating the necessity for reactions from the VSOC. A man-in-the-middle attack can disrupt communication between the VSOC and ECU, thereby halting the reaction process. Such vulnerabilities can be mitigated by establishing an internal reaction system.

Acknowledgment

This paper is supported by the project IDPS at Bosch Business sector Mobility (BBM) and by the European Union-funded project CyberSecDome (Agreement 101120779).

References

[1] M. Hamad, M. Tsantekidis, and V. Prevelakis, "Intrusion response system for vehicles: Challenges and vision," in *International Conference on Smart Cities and Green ICT Systems*, pp. 321–341, Springer, 2019.

[2] United Nations Economic Commission for Europe, "UN Regulation No. 155: Cybersecurity and Cybersecurity Management Systems for Vehicles," 2020.

[3] AUTOSAR, *Specification of Intrusion Detection System Manager*, 2020.

[4] "AUTOSAR SWS:Specification of Intrusion Detection System Manager," 2020.

[5] M. Hamad, M. Tsantekidis, and V. Prevelakis, "Red-zone: Towards an intrusion response framework for intra-vehicle system," in *Proceedings of the 5th International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS)*, pp. 148–158, 2019.

[6] M.-K. Yoon, S. Mohan, J. Choi, J.-E. Kim, and L. Sha, "Securecore: A multicore-based intrusion detection architecture for real-time embedded systems," in *2013 IEEE 19th Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pp. 21–32, 2013.

[7] M. Ring, D. Frkat, and M. Schmiedecker, "Cybersecurity evaluation of automotive e/e architectures," in *ACM Computer Science In Cars Symposium (CSCS 2018)*, 2018.

[8] C. Miller and C. Valasek, "Adventures in automotive networks and control units," *Def Con*, vol. 21, no. 260-264, pp. 15–31, 2013.

[9] S. Nie, L. Liu, and Y. Du, "Free-Fall: Hacking Tesla from Wireless to CAN Bus," in *Black Hat Europe*, pp. 1–22, 2018.

[10] M. Wolf, R. Lambert, A.-D. Schmidt, and T. Enderle, "Wanna drive? feasible attack paths and effective protection against ransomware in modern vehicles," 2017.

[11] "Cyber Kill Chain : Intelligence Driven Defense," tech. rep., Lockheed Martin Corporation, 2014. Online; accessed on March 2, 2023.

[12] J. Straub, "Modeling attack, defense and threat trees and the cyber kill chain, att&ck and stride frameworks as blackboard architecture networks," in *2020 IEEE International Conference on Smart Cloud (SmartCloud)*, pp. 148–153, 2020.

[13] M. Mackay, "When 21434 and R155 ATTACK." https://fnc.itu.int/wp-content/uploads/2021/03/Matthew-Mackay_GM-FNC-2021.pdf.

[14] The MITRE Corporation, "D3FEND." https://d3fend.mitre.org/. Online; accessed on March 30, 2023.

[15] "Experimental Security Assessment of BMW Cars: A Summary Report," tech. rep., Tencent Security Keen Lab, 2018.

[16] "Experimental security assessment of mercedes-benz cars," tech. rep., Tencent Security Keen Lab, 2021.

[17] V. Prevelakis and M. Hamad, "A policy-based communications architecture for vehicles," in *2015 International Conference on Information Systems Security and Privacy (ICISSP)*, pp. 155–162, IEEE, 2015.

[18] "AUTOSAR SWS: Specification of Execution Management," AUTOSAR Adaptive Platform Specification: R20-11, AUTOSAR Development Partnership, 2020.

[19] K. Scarfone and P. Mell, "Guide to intrusion detection and prevention systems (idps)," Special Publication 800-94, National Institute of Standards and Technology, 2010.

[20] "Continuous Security Monitoring with ESCRYPT Intrusion Detection Solutions." https://www.etas.com/download-center-files/DLC_products_ESCRYPT/etas-flyer-escrypt-cycurids-en-20221220.pdf, 2022.

[21] M. Hamad, M. Nolte, and V. Prevelakis, "A framework for policy based secure intra vehicle communication," in *2017 IEEE Vehicular Networking Conference (VNC)*, pp. 1–8, IEEE, 2017.

[22] "Road vehicles — Unified diagnostic services (UDS) — Part 1: Specification and requirements," Standard ISO 14229-1:2020, International Organization for Standardization, Geneva, CH, 2020.

[23] M. Tsantekidis, S. Abdelghani, M. Hamad, and V. Prevelakis, "Creating a security enforcement environment for a vehicular platform," in *2023 IEEE Conference on Standards for Communications and Networking (CSCN)*, pp. 278–283, 2023.

[24] B. Potteiger, Z. Zhang, L. Cheng, and X. Koutsoukos, "A tutorial on moving target defense approaches within automotive cyber-physical systems," *Frontiers in Future Transportation*, vol. 2, 2022.

[25] Microsoft Corporation, "Microsoft Threat Modeling Tool 2016." https://www.microsoft.com/en-us/download/details.aspx?id=49168, 2016. Online; accessed on March 29, 2023.

[26] T. Hoppe, S. Kiltz, and J. Dittmann, "Applying intrusion detection to automotive it-early insights and remaining challenges," *Journal of Information Assurance and Security (JIAS)*, 2009.

[27] S.-W. Lee, Y. Yoon, and H. K. Lee, "Practical vulnerability-information-sharing architecture for automotive SRA," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 9, pp. 3669–3680, 2019.

[28] M. Roesch, "Snort: Lightweight intrusion detection for networks.," in *LISA*, pp. 229–238, USENIX, 1999.

[29] A. Wasicek, M. D. Pesé, A. Weimerskirch, Y. Burakova, and K. Singh, "Context-aware intrusion detection in automotive control systems," in *Proc. 5th ESCAR USA Conf*, pp. 21–22, 2017.

[30] J. Jiang, C. Wang, S. Chattopadhyay, and W. Zhang, "Road context-aware intrusion detection system for autonomous cars," in *Information and Communications Security*, Springer International Publishing, 2020.

[31] S.-F. Lokman, A. T. Othman, and M.-H. Abu-Bakar, "Intrusion detection system for automotive controller area network (can) bus system: a review," *EURASIP Journal on Wireless Communications and Networking*, vol. 2019, pp. 1–17, 2019.

[32] "Security software for Automotive System-on-Chip ESCRYPT CycurSoC." https://etas-cn.net/download-center-files/DLC_products_ESCRYPT/etas-flyer-escrypt-cycursoc-en-20221220.pdf. Online; accessed on March 23, 2023.