REACT: Autonomous Intrusion Response System for Intelligent Vehicles

Mohammad Hamad[a,*], Andreas Finkenzeller[a], Michael Kühr[a], Andrew Roberts[b], Olaf Maennel[c], Vassilis Prevelakis[d], Sebastian Steinhorst[a]

[a]*Technical University of Munich, Munich, Germany*
[b]*Tallinn University of Technology, Tallinn, Estonia*
[c]*University of Adelaide, Adelaide, Australia*
[d]*Technical University of Braunschweig, Braunschweig, Germany*

Abstract

Autonomous and connected vehicles are rapidly evolving, integrating numerous technologies and software. This progress, however, has made them appealing targets for cybersecurity attacks. As the risk of cyber threats escalates with this advancement, the focus is shifting from solely preventing these attacks to also mitigating their impact. Current solutions rely on vehicle security operation centers, where attack information is analyzed before deciding on a response strategy. However, this process can be time-consuming and faces scalability challenges, along with other issues stemming from vehicle connectivity. This paper proposes a dynamic intrusion response system integrated within the vehicle. This system enables the vehicle to respond to a variety of incidents almost instantly, thereby reducing the need for interaction with the vehicle security operation center. The system offers a comprehensive list of potential responses, a methodology for response evaluation, and various response selection methods. The proposed solution was implemented on an embedded platform. Two distinct cyberattack use cases served as the basis for evaluating the system. The evaluation highlights the system's adaptability, its ability to respond swiftly, its minimal memory footprint, and its capacity for dynamic system parameter adjustments. The proposed solution underscores the necessity and feasibility of incorporating dynamic response mechanisms in smart vehicles. This is a crucial factor in ensuring the safety and resilience of future smart mobility.

Keywords:
Security, Intrusion Response System, Autonomous Vehicle

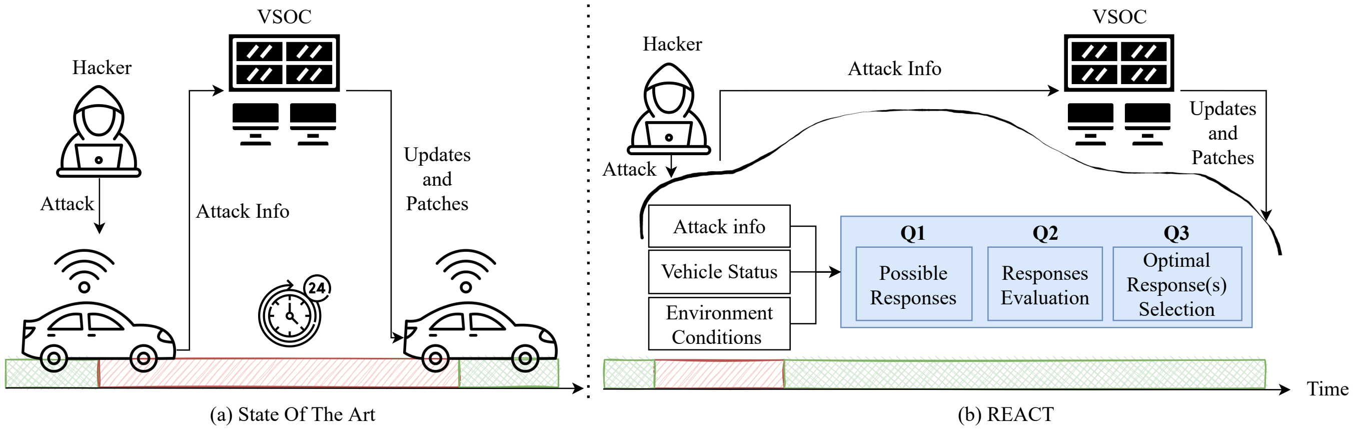
1. Introduction

In recent years, there has been remarkable progress in the development of smart vehicles. Today's vehicles resemble interconnected networks on wheels, with numerous embedded computers, called Electronic Control Units (ECUs), linked through various types of networks, hosting an extensive number of software components totaling over a hundred million lines of code. Moreover, these networks incorporate various intelligent sensors (such as cameras, LiDAR, radar, etc.) and different connectivity technologies that enhance the vehicle's ability to perceive and interact with the surrounding environment, thus bolstering autonomy and minimizing the reliance on human intervention. However, with the rise of connectivity and the softwarization of vehicles, the vulnerability to cyberattacks targeting these systems has also escalated [66].

Recently, there has been a growing interest in addressing the security threats that may target smart vehicles. For instance, the ISO 21434 [34] standard has been introduced, with a significant portion dedicated to the development of threat analysis and risk assessment methodologies. Moreover, the field of intrusion detection and prevention in the automotive domain has witnessed extensive research, leading to various avenues for research [39]. However, despite these efforts, the number of attacks targeting smart vehicles continues to rise [66]. This is to be expected, as security is not absolute, and we must acknowledge that complete prevention of all security threats may not be attainable. Therefore, greater emphasis should be placed on defining *how the system should behave when confronted with such unavoidable attacks.*

The cybersecurity incident response is an integral aspect of security management, as outlined in ISO/SAE 21434 within the operational and maintenance clause [34]. Based on the standard, this process aims to provide remedial actions and updates, which may involve post-development changes to address security vulnerabilities. The process necessitates the vehicle to share cybersecurity information about the vulnerability that triggered the cybersecurity incident response. Being part of the ISO/SAE 21434, it is now imperative that manufacturers comply with new regulations by having a cybersecurity management system that oversees the cybersecurity activities and processes in the product life-cycle. To achieve this, Vehicle

[*] Corresponding author.
Email addresses: mohammad.hamad@tum.de (Mohammad Hamad), andreas.finkenzeller@tum.de (Andreas Finkenzeller), michael.kuehr@tum.de (Michael Kühr), andrew.Roberts@taltech.ee (Andrew Roberts), olaf.maennel@adelaide.edu.au (Olaf Maennel), prevelakis@ida.ing.tu-bs.de (Vassilis Prevelakis), sebastian.steinhorst@tum.de (Sebastian Steinhorst)



(a) State Of The Art

(b) REACT

Figure 1: On the left side, the current vehicle system shares attack information with the VSOC but often has to wait for extended periods to receive necessary security patches and updates. This waiting period puts the vehicle in a malicious status (red, diagonal lines). On the right side, the vehicle can select and implement security solutions to avoid the long waiting time for security patches and updates and return to normal status (green, cross diagonal lines).

Security Operation Centers (VSOCs) will be utilized to support monitoring [6, 59, 51]. Such VSOCs will employ expert teams that continuously analyze data collected from all connected vehicles, enabling automakers to swiftly and efficiently address security incidents [51]. Although it's arguable that numerous tasks within a VSOC could be automated, the challenge of scalability persists, especially considering the extensive fleet of connected vehicles and the immense data volumes accumulated by each vehicle, reaching terabytes [70]. The transfer and processing of such data turn out to be significant issues, particularly in urban areas with hundreds of cars per vicinity, leading to bottlenecks. Additionally, the connectivity itself could be an attractive target for attackers. Also, the integration of VSOCs into the smart vehicle ecosystem demands solutions for addressing connectivity challenges between vehicles and the VSOC, as well as managing privacy concerns tied to shared data [22].

Finally, and more importantly, there is a need to ensure a near-real-time response to security attacks. Taking into account the need for a human in the loop, as well as the latency introduced by high-volume shared data and communication between the vehicles and the VSOC, achieving a near-real-time response seems unrealistic. This perspective is supported by the European Union Agency for Cybersecurity (ENISA), which has cautioned that responding to high-criticality attacks could potentially take days or even weeks [15]. The scenario of extended waiting presents a dilemma, with two options, each having its own disadvantages. Allowing a vehicle to operate with a compromised component due to extended waiting for a security update is far from the ideal situation. Alternatively, suspending the compromised component until the security update is received might not be the best course of action either, particularly if the component plays a crucial role in operations.

Contributions: Therefore, there is a need for vehicles to be equipped with the capability to swiftly respond to cyberattacks. However, having such an capability requires the answering of three main questions (see Figure 1): **Q1:** What are the possible responses that can be taken? **Q2:** What factors need to be considered when evaluating these responses? **Q3:** How to select one or more of these response during the run-time based on the responses evaluation? This paper aims to address these questions by investigating and categorizing potential responses according to the impact of various cyber attacks to which each response aims to react. Additionally, the paper presents a dynamic risk assessment and cost evaluation for attacks and responses, utilizing given data such as attack information and vehicle status. This assessment supports the selection of suitable responses. Furthermore, the paper explores different approaches for response selection, conducts comparisons, and identifies those best suited for automotive systems. Lastly, the paper introduces an incident response system, evaluates it using two attack scenarios, and discusses both the quality of the responses it generates and its overall efficiency. In summary, the main contributions of this paper are as follows:

- We conduct a comprehensive review of existing intrusion response strategies for IT systems and mapped them to automotive systems, considering the unique characteristics of automotive attacks and automotive system architectures (see § 2).
- We proposed a novel method for calculating the cost and response benefits by extending existing risk assessment approaches specific to automotive systems (see § 3).
- We explore a range of algorithms for selecting appropriated responses, conducted comparative analyses, and identified the most suitable algorithms for automotive systems, proposing their adoption to enhance automotive security (see § 4).
- We introduced the first automotive Intrusion Response System (IRS) and provided an open-source prototype[1] (see § 5).

[1] https://github.com/mohammadhamad/REACT

- We demonstrat the feasibility and applicability of the proposed automotive IRS through evaluations using embedded resource platforms and two attack scenarios. Findings indicated that the system can adapt to different scenarios, make response selections quickly (average 30 ms for the worst-case algorithm), have low memory overhead, and dynamically adjust system parameters (see § 6).

2. Response Strategies

The purpose of this section is to address the first question (**Q1**) about possible response strategies. To do so, it is critical to have a deep understanding of the system as well as the potential attacks and threats it may face. Therefore, this section introduces the design of an automotive reference architecture, discusses the potential threats that may arise, and provides a comprehensive summary of the different response strategies that can be utilized to mitigate these attacks.

2.1. Automotive Reference Architecture

In order to understand how IRS can be integrated into modern vehicles and the potential responses they can provide, it is essential to first understand their system architecture. Figure 2 presents a generic, realistic and comprehensive reference architecture that can be found in modern vehicles. It is notable that a modern vehicle includes *highly interconnected* subsystems. The figure also shows how modern vehicles have many *embedded devices*, know as ECUs, which are *distributed* allover the vehicle, communicating among themselves via different types of networks such as CAN, Flexray and Ethernet. These ECUs are grouped in different domains or zones based on the functionality such as infotainment, Advanced Driver Assistance Systems (ADAS), powertrains, etc. Besides ECUs, modern vehicles are equipped with many sensors (e.g., cameras, LiDAR, etc.), advanced communication technology for connecting with the external world, and diagnostic ports (e.g., OBD-II) that collectively form a significant attack surface for different types of attacks and threats [11]. The unrestricted or/and uncontrolled interaction among all those components puts the whole system in danger. Attackers could launch a *stepping-stone* attack [65], where they compromise a non-critical ECU with weaker security (e.g., the infotainment system), in order to gain control of a more crucial one (e.g., engine control).All these characteristics of the vehicle architecture suggest that any proposed IRS should take into account the constrained resources and the highly interconnected and distributed nature of a vehicular system.

2.2. Threats and Attacks

Threat Analysis and Risk Assessment (TARA), an essential component of ISO 21434, is employed as a systematic way to identify and assess cybersecurity threats and risks in the automotive industry, facilitating the implementation of effective mitigation strategies. Since TARA does not dictate a specific method to identify threats, various methods have been proposed, such as STRIDE [37], SAVTA [21], attack trees [26, 20], and many others [45]. Following the methodology of TARA, these methods provide a comprehensive list of threats and attacks that may target the vehicular system and offer preventive measures. However, they do not address the reactive measures required for an automotive IRS.

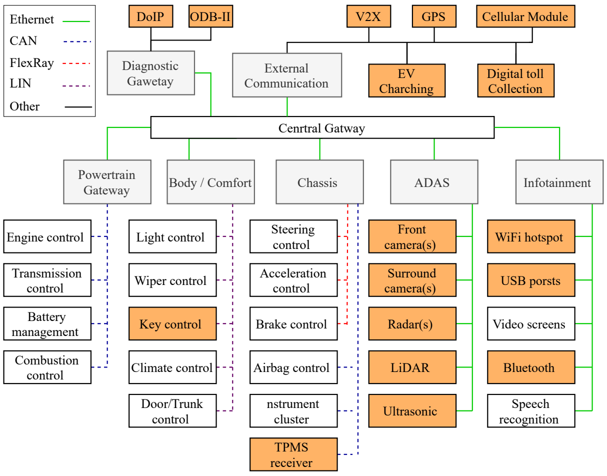


Figure 2: Reference vehicle architecture with possible attack surface (orange).

Using the list of threats and attacks to create a response for each of them seems to be not ideal due to several challenges, including the large number of attacks and the requirements for precise information about each attack, which must be provided by the Intrusion Detection System (IDS). This challenge becomes evident when considering Zero-Day attacks, where information about such attacks may not be available to the IRS at the time of detection by the IDS. Even if an anomaly-based IDS shares some information about the attack pattern with the IRS, a response solely based on known attack patterns may not sufficiently react to these Zero-Day attacks. Therefore, the most effective approach is to enable the IRS to understand the situation it aims to respond to. This involves focusing on the impact or outcome of different attacks rather than solely on the attacks themselves.

To achieve that, we have developed a model, illustrated in Figure 3, which represents the actual results of intrusions collected from various research works. The model encompasses five main attack outcomes, each of which can result from multiple types of attacks. Examples of these attacks are depicted in the outer nodes of Figure 3. Also, to reflect the outcome of stepping-stone attacks, the model links the different outcomes to demonstrate that certain attacks may cause series of results. The five attack outcomes are:

- *Falsify / Alter Information:* Different attacks have the potential to modify information on a bus or within an ECU. It is important to note that not every alteration of information automatically results in undesirable behavior. For instance, adversarial samples [46], such as incorrect classifications of objects detected by a camera, may not necessarily lead to incorrect behaviors.

- *Falsify / Alter Timing:* This outcome typically occurs as a

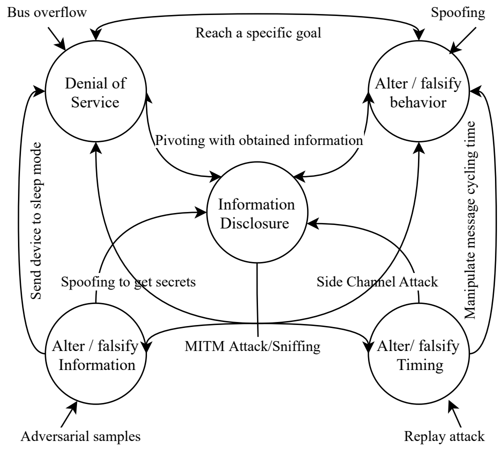


Figure 3: Classification of intrusion results and examples of attacks for each possible intrusion result.

result of attacks targeting the communication buses of the vehicle [69, 43] or the real-time tasks on the ECUs [19].

- *Information Disclosure:* This outcome is the result of attacks, such as spoofing, eavesdropping, and others, that aim to allow attackers to gain unauthorized access to sensitive information exchanged during communication or stored within the ECUs [13].

- *System Unavailability:* This outcome typically occurs as a result of Denial of Service (DoS) attacks that aim to cause a loss of availability for a specific component or subsystem in the vehicle [53]. Such attacks can lead to severe damage to the system, especially if they target high-critical components [1].

- *Falsify / Alter behavior:* This outcome is the result of tampering attacks that specifically target the components, data, or parameters of a system with the intention of altering the system's intended behavior and achieving unauthorized or malicious outcomes [47]. While this intrusion outcome may appear similar to falsify/alter information, the key distinction is that in falsify/alter information attacks, the goal is to tamper with the information itself without the explicit method of changing the system's behavior, even though it may indirectly lead to such changes.

2.3. Response Possibilities

After classifying the outcome of the attack, it becomes easier to determine which responses can be used to address that particular outcome and handle the attacks that cause it. In order to do so, we have examined typical responses discussed in both the automotive and non-automotive domains. It should be noted that while some research papers in the automotive domain have discussed the need for responses to certain attacks, there is currently no comprehensive research that lists and classifies all possible responses. Furthermore, it is important to consider that some of the responses we collected were originally designed for computer networks and may not be directly applicable to automotive bus systems due to the lack of specific security mechanisms [14]. For example, response actions such as IP address changes or port blocking [4] are highly specific to Ethernet and higher protocols such as IP, and therefore have limited suitability for certain aspects of communication in vehicles. To address this challenge, we have defined a list of generic responses that are specific enough to be applied in an automotive IRS, while also being adaptable to constrained and potentially insecure devices. Table 1 provides an overview of the different responses based on the identified attack outcomes. In addition, we have included a "General" category that encompasses responses applicable to all five categories. For more detailed information about each response, please refer to the respective sources cited in Table 1.

3. Dynamic Cost and Impact Evaluation

In this section, we will address **Q2** by outlining the key factors required to enable the selection of the most effective response by the IRS. These factors can be categorized into two groups: *intrusion-related factors*, which pertain to the attack's impact and risk, and *response-related factors*, which concern the cost and benefit of the chosen response.

3.1. Intrusion-Related Factors

3.1.1. Intrusion Properties

For each detected intrusion, the following properties need to be determined:

- *Source of the intrusion:* This represents the component from which the attack was launched. Referring to the automotive reference architecture depicted in Figure 2, sources can include entities from the attack surface as well as external attackers targeting any of these components.

- *Destination of the intrusion:* The attacked entity can be described as the destination of the intrusion. This could be ECUs, sensors, or bus systems.

- *Intrusion result:* This refers to one of the outcomes that were previously defined in Subsection 2.2. Similar to the source and destination of an intrusion, this information is also provided by an IDS.

- *Intrusion Impact*: This information serves to depict the impact of the intrusion on the system and is essential for evaluating the risks during the attack.

3.1.2. Dynamic Attack Impact Assessment

To assess the potential risks associated with an intrusion, it is necessary to understand the impact of the attack and the likelihood of its occurrence [34, 42]. To calculate the impact of the intrusion, many methods were already adopted such as HEAVENS [35]. HEAVENS classifies the impact of a given threat based on four metrics [68, 45]:

1. Safety impact, denoted as S with $S \in \{0, 10, 100, 1000\}$

Table 1: Classification of generic responses to intrusion results.

Intrusion Result	**Response Index. Response**
Falsify / Alter Timing	**1**. Use of redundant information [24], **2**. Correction of timing [54, 14], **3**. Force additional authentication [4], **4**. Restart the device/system [38], **5**. Change settings [30], **6**. Redirect traffic [30], **7**. Re-initialization [27]
Falsify / Alter Information	**1**. Use of redundant information (Reallocation) [24], **3**. Force additional authentication [4], **4**. Restart the device/system [38], **8**. Create a backup [12], **5**. Change settings [30], **7**. Re-initialization [27], **9**. Correct protocol specification faults [28], **10**. Split or merge functions [72]
Information Disclosure	**11**. Issue authentication challenges [54], **12**. Re-enforce access control [2], **3**. Force additional authentication [4], **13**. Introduce a honeypot [2], **4**. Restart the device/system [38], **14**. Modify firewall [30], **6**. Redirect traffic [30], **10**. Split or merge functions [72], **7**. Re-initialization [27], **15**. Network isolation [14]
System Unavailability	**1**. Use of redundant information (Reallocation) [24], **12**. Re-enforce access control [2], **13**. Introduce a honeypot [2], **4**. Restart the device/system (source or destination) [38], **14**. Modify firewall [30], **6**. Redirect traffic [30], **10**. Split or merge functions [72], **7**. Re-initialization [27], **16**. Limit resources of the attacker [12], **17**. Safe mode [23]
Falsify / Alter Behavior	**1**. Use of redundant information (Reallocation) [24], **18**. Correction of behavior [54], **9**. Correct protocol specification faults [28], **3**. Force additional authentication [4], **19**. Restart the miss-behaving system [38], **5**. Change settings [30], **10**. Split or merge functions [72], **7**. Re-initialization of the miss-behaving device [27], **17**. Safe mode [23], **8**. Create a backup [12]
General	**20**. Isolation [24], **21**. Limit communication of malicious system [24], **22**. Drop packets [38], **23**. Trace communication [24], **24**. Introduce additional logging [4], **25**. Block network traffic [2], **26**. Kill process [24], **27**. Reduce trust level of the source [24], **28**. Perform a security auditing [23], **29**. Request / Perform software update [54], **30**. Notify Security Operations Center (SOC) / administrator [3, 2], **31**. No action [3], **32**. Adapt parameters for IDS [25], **33**. Warn / inform other ECUs [5, 24]

2. Financial impact, denoted as F with $F \in \{0, 10, 100, 1000\}$
3. Operational impact, denoted as O with $O \in \{0, 1, 10, 100\}$
4. Privacy impact, denoted as P with $P \in \{0, 1, 10, 100\}$

In the original HEAVENS method, the overall impact I is calculated as a sum of the four single impacts as depicted in Equation 1 [68].

$$I = S + F + O + P \tag{1}$$

One issue with the impact calculation, as presented in Equation 1, is the overemphasis on safety and financial parameters. This skewed emphasis not only complicates the comparison and independent evaluation of the four metrics but also renders it unsuitable for an automotive IRS. In the automotive context, safety and operational considerations typically outweigh financial and privacy-related aspects for most automotive functions. Considering the aforementioned issue, we propose normalizing all possible values to $0, 1, 10, 100$, representing no, low, medium, or high impact for each of the four metrics in HEAVENS.

Another limitation of the current risk assessment method, including HEAVENS, is its failure to account for dynamic environmental factors, such as run-time context, operational status, and the surrounding environment. This gap may arise because HEAVENS is primarily applied during the design phase, making it somewhat oblivious to run-time conditions. To address this challenge and enhance the method's applicability for use within automotive IRS, we introduce a new metric termed "Environment," denoted as E. This metric, E, encompasses dynamic factors that are crucial for assessing intrusion impact [24]. Potential inputs that can be used to derive the environmental parameter E include vehicle speed, road conditions, the proximity of nearby objects, and more. These parameters can exert significant influence, as a single intrusion may yield different impacts depending on physical and environmental considerations.

The final enhancement option for the HEAVENS method involves the capability to dynamically adjust the assessment of intrusion impact. Following a successful intrusion response, it may become evident that the stored parameters for S, F, O, P, and E require a different representation. HEAVENS currently confines impact values to $0, 1, 10, 100$, and a simple adjustment to a new value could result in significant over-representation. To address this issue, introducing weights for each of the five evaluation metrics (w_S, w_F, w_O, w_P, and w_E) offers a valuable mechanism for accommodating learning and adaptation processes. The optimization proposals discussed earlier to transform the calculation of intrusion impact using the HEAVENS method into a dynamic process lead to the Equation 2.

$$I = w_S \cdot S + w_F \cdot F + w_O \cdot O + w_P \cdot P + w_E \cdot E \tag{2}$$

Utilizing dynamically adjusted static values for S, F, O, and P, each incorporating their respective weights, in addition to dynamically acquired values for E along with an adapted static weight. In cases involving specific automotive architectures, the equation can also be applied in a more granular fashion for particular assets. Initial values for all these parameters can be established by security experts, drawing upon their experiential knowledge.

The source and destination of the attack are employed to determine the attack's location, aiding in the calculation of the subsequent attack likelihood, especially when considering stepstone attacks, across various parts of the system. This assessment of attack likelihood, in conjunction with the evaluation of attack impact, contributes to the overall risk assessment.

3.2. Response-Related Factors

3.2.1. Response Properties

Similar to the intrusion, each response will have five properties that need to be identified:

- *Actual action:* They refer to the actual actions taken in the event of an intrusion. These actions can be selected from those presented in Table 1.
- *Precondition:* Some responses may require preconditions that must be met. These preconditions can be expressed as Boolean expressions and serve as prerequisites to trigger the response.
- *Place of application:* Refers to the location where the response will be implemented. A response can be applied either at the source entity of an intrusion, the destination, or at both locations.
- *Stop condition:* Refers to the condition for which the implemented response should cease. This condition can be related to a specific time [44], the successful reestablishment of security policies [24], or the necessity for persistent measures [65].
- *Cost and benefit of the response:* Refers to the costs and benefits incurred when implementing a response to an intrusion or security incident.

3.2.2. Dynamic response cost and benefit assessment

When considering the cost of responses, various methods were employed to determine their value in IT systems [60]. These methods primarily rely on one of three models: a static cost model that assigns a fixed cost value for each response, a static evaluated cost model that calculates cost using a static function with some adjustment possibilities, or dynamic evaluated cost models that offer fully dynamic evaluation based on real-time data. Each model varies in terms of simplicity, adaptability, and accuracy, catering to different system requirements and scenarios.

Statically evaluated cost models provide a valid trade-off between achievable implementation efforts, especially on constrained devices similar to the one used in automotive system, and plausible results. These models maintain a static approach to calculating response costs, even though the actual cost values may vary. Various metrics for calculating response costs are mentioned in current literature. The first metric evaluates the impact of the response on availability [60]. Availability's impact is represented as $A \in 0, 1, 10, 100$ to ensure consistency with intrusion metrics. The second metric, describing the response cost, assesses its effect on the performance of the (sub)system [60], similar to the deployment cost of countermeasures [18]. This metric is denoted as $Perf \in 0, 1, 10, 100$ to maintain a uniform scale with the impact of the response on availability.

To achieve results similar to the adapted HEAVENS method described in § 3.1, a comparable equation can be employed to calculate the cost (c) of a response. By adopting specific weights (w_A and w_{Perf}) for the impact on availability and performance along with their actual values (A and $Perf$), the response cost can be computed as shown in Equation 3. This approach results in a highly adaptable method for calculating the response cost. While the initial values for A and $Perf$ can be manually determined, they can also be adjusted over time. The specific weights offer a means to introduce a learning component within the mathematical framework.

$$c = w_A \cdot A + w_{Perf} \cdot Perf \tag{3}$$

Likewise, the adapted HEAVENS method introduced in § 3.1 can be repurposed for evaluating the benefit of a response, with the exception of the environmental parameter E and its associated weight w_E. While HEAVENS assesses intrusion impact using four metrics, these same metrics can be employed to quantify the benefits in these four categories when assessing response value. By employing identical value possibilities with $S, F, O, P \in 0, 1, 10, 100$, a corresponding benefit value can be determined. The calculation of the benefit (b) for each response option, as shown in Equation 4, is derived from Equation 2.

$$b = w_S \cdot S + w_F \cdot F + w_O \cdot O + w_P \cdot P \tag{4}$$

For each response option classified in Table 1, the cost calculated using Equation 3 and the benefit determined using Equation 4 must be applied, and preconditions must be established. Initial values for S, F, O, P, A, and $Perf$, along with their respective weights, can be assigned by security experts and subsequently updated either manually or through learning algorithms within an IRS. Similar to the impact calculation of intrusions, these weights can be adjusted to improve the accuracy of the model.

4. Optimal Selection Algorithms

In this section, we will address the third question **Q3**, by exploring numerous potential methods for selecting response strategies (§ 4.1), compare these approaches and provide a rationale for our chosen strategy (§ 4.2), and describe how to adopt the selected strategies (§ 4.3).

4.1. Possible Algorithms

To determine the best method for selecting appropriate responses, we explore various algorithms and solutions used in *non-automotive domains* and compare them to identify the most suitable one that can be implemented within the vehicle system. Several surveys, such as [50, 7, 8], provide valuable insights into response selection approaches in non-automotive domains, making them worth investigating for more comprehensive details.

4.1.1. Simple Additive Weighting (SAW)

SAW [17] is the simplest and most often used method. The basic concept of this method is to find a preference value (p) for each possible response, and then select the response with the highest preference value as the best option. To illustrate how this method works, let us assume that we have n possible responses ($\mathcal{R} = \{r_1, r_2, \ldots, r_n\}$) and m criteria ($C\mathcal{R} = \{cr_1, cr_2, \ldots, cr_m\}$) that will be used as a reference for evaluating the responses. Each criterion will be assigned a weight w_j where $\sum_{j=1}^{m} w_j = 1$. To calculate the preference values, a normalized decision matrix is first created, where each element of the matrix is normalized based on the nature of the criterion, whether it is a cost or benefit, as shown in Equation 5.

$$\alpha_{ij} = \begin{cases} \frac{v_{i,j}}{\max_i(v_{i,j})}, & \text{if criterion } cr_j \text{ is a benefit} \\ \frac{\min_i(v_{i,j})}{v_{i,j}}, & \text{if criterion } cr_j \text{ is a cost} \end{cases} \tag{5}$$

where $v_{i,j}$ is the performance value of the response r_i when it is evaluated in terms of criterion cr_j. The preference value (p_i) of response r_i is then obtained by calculating the weighted sum of the normalized performance values using Equation 6.

$$p_i = \sum_{j=1}^{m} w_j \cdot \alpha_{ij} \tag{6}$$

Finally, the response r_i with the highest preference value (p_i) is considered as the best selection response.

4.1.2. Linear Programming (LP)

LP is a mathematical technique that can be employed to select optimal responses [29]. LP can be used to find the best combination of responses that *maximizes* or *minimizes* a certain objective function. To illustrate the workings of this method, let's consider a scenario where we have n possible responses ($\mathcal{R} = r_1, r_2, \ldots, r_n$). The optimization of the objective function can be as in Equation 7.

$$\sum_{i=1}^{n} x_i s_i \rightarrow \max \textit{or} \min \tag{7}$$

where x_i represent a criteria related to the response r_i and $\overrightarrow{s}$ be a vector of binary decision variables, where s_i is equal to 1 , it indicates that the corresponding response $r_i \in \mathcal{R}$ will be executed. Conversely, if s_i is equal to 0 , it signifies that the response $r_i \in \mathcal{R}$ will not be executed. The optimization problem typically includes *constraints* to ensure the selection process adheres to specific conditions or limitations.

4.1.3. Game-Theoretic Algorithm

Another mathematical method to determine optimal responses against cyber attacks is game-theoretic algorithms [72, 73, 67]. In the game-theoretic approach, the attacker and the IRS are modeled as two players. Each player has a set of actions available to them, such as different attack strategies $\mathcal{A} = \{a_1, a_2, \ldots, a_k\}$ for the attacker and response strategies $\mathcal{R} = \{r_1, r_2, \ldots, r_n\}$ for the IRS. The goal of the IRS is to select the optimal response to the attack at a given time. One way to achieve that is by minimizing the maximum damage of the attack: $\min_{r_i \in \mathcal{R}}(\max_{a_i \in \mathcal{A}}(U(r_i, a_i)))$ where $U(r_i, a_i)$ represents the utility function for the IRS when the attacker chooses attack a_i and the IRS responds with response r_i.

4.1.4. AI-based mechanisms

Many AI-based mechanisms were used to support the dynamic selection of the response such as Genetic Algorithms [16], Convolutional Neural Networks [71], Supervised machine learning [61], Q-Learning [33], and many more [57]. Using any of these AI models usually requires many steps including data collection and pre-processing, feature extracting, model training, and feedback loop to improve the quality of the selected responses.

4.1.5. Other Methods

There are alternative mathematical approaches to IRSs that are not derived from general mathematical problems. One example is REASSESS [52] that uses human-evaluated metrics and prior responses to select optimal responses. While it offers simplicity, this reliance on human evaluation can lead to inaccurate assumptions. Its mandatory learning behavior is unsuitable for automotive systems, and it lacks the option for flexible learning to enhance responses, requiring a well-established feedback loop. Another simpler approach is the cost-sensitive generic framework [62, 63], which includes steps like defining operational costs, ranking responses using a weighted sum method, and selecting the best response with an intrusion matrix. However, its reliance on static value assignments and sensitive parameters, typically defined by human experts, can make objective assessment challenging and result in potentially harmful responses.

4.2. Comparison

Table 2 summarizes all the advantages and the drawbacks of the five classes of response selection algorithms.

The primary advantage of SAW is its relative simplicity and utilization of lightweight mathematical operators, making it suitable for running on constrained devices with a polynomial run-time, without requiring complex external libraries [9]. However, the main drawback of SAW is the need for an adapted SAW method to achieve more accurate results. This often leads to increased complexity and longer run-time compared to the original SAW. Another drawback is the dependency on subjective parameters such as specific weights. This dependency can result in highly variable outcomes that may not accurately reflect the system state [41].

Table 2: Comparison of the different response selection methods

Method	Benefits	Drawbacks
SAW	+ Simplicity and lightweight operators + Suitable for constrained devices + Polynomial run-time	- Adapted methods for accuracy increase complexity - Reliance on subjective parameters
LP	+ Flexible structures + Typically polynomial run-time + Existing libraries for solvers	- Higher complexity for modeling and calculation - Theoretically exponential run-time
Game-Theoretic Algorithms	+ System state consideration + Accurate system representation	- Very complex models - Computational complexity - Reliance on subjective parameters
AI-based Solutions	+ Handle large amount of data + Fast response selection	- Uncertainty of the selected responses - High resource requirements
Other Methods	+ Simple mathematical models + Typically fast + Combination with other methods possible + Learning is possible	- Complexity raises with large systems - Human influence has always subjective opinions

A major benefit of LP is its ability to formulate a single objective function and multiple constraints, providing an accurate representation of multi-objective optimization problems. However, compared to SAW, LP requires complex implementation, resulting in increased computational complexity for large systems [29]. The run-time of the algorithm depends on the solving method employed, such as the commonly used Simplex algorithm. While the Simplex algorithm has polynomial run-time for "typical" problems [58], it exhibits exponential worst-case run-time in theory [40].

The advantage of game-theoretic approaches lies in their consideration of the system state, resulting in a highly accurate representation of the system. Furthermore, game-theoretic approaches can be deployed in a distributed manner, as highlighted in [73]. A major drawback of this method is the use of highly complex models, which are necessary to determine optimal moves in game-theoretic algorithms. Solving such complex models often requires significant resources and leads to large communication overhead [73], making this approach unsuitable for constrained devices. Additionally, most models in practice make assumptions or simplifications due to the near-infinite number of possible system states [72, 73, 67], as complete modeling of all states is infeasible.

Using AI-based methods is still limited because of many issues such as the high memory and computation requirements of some of these methods [31] and the unrealistic responses that some models can produce (e.g., Genetic Algorithms). Additionally, uncertainty surrounding the outputs of these models limits their adoption. Finally, most of these methods rely on the availability of datasets for model training. However, autonomous vehicles often operate in dynamic and unpredictable environments. When the operating environment significantly deviates from what the AI has learned, it may encounter challenges in adapting effectively or making appropriate decisions.

Finally, while the cost-sensitive generic framework and RE-ASSESS are simple and demonstrate promising in computer and network technologies, adapting them to a highly heterogeneous multi-bus architecture, like the vehicular reference architecture, presents significant challenges.

Considering the factors discussed above, we have chosen to explore the adapted SAW method, as well as LP with a focus on both benefit maximization and cost minimization for the design of an automotive IRS. The remaining algorithm families were assessed but are not pursued further, as explained earlier.

4.3. Adopting of SAW and LP

4.3.1. Adopting of SAW

To adopt the SAW method for automotive IRSs, we first need to define the criteria $\mathcal{CR}$ that will be used to evaluate each response. For this purpose, we can utilize the HEAVENS parameters, including the cost of a response c (see Equations 3) and the benefit of a response b (see Equation 4). However, using these two parameters still presents some issues that need to be addressed in order to effectively use and adapt SAW for valid results. The first problem arises when using these parameters during the creation of the elements of the normalized decision matrix, as depicted in Equation 5. This problem originates from the fact that our modified HEAVENS method allows values of $v_{i,j}$ to be in the set $0, 1, 10, 100$ for both criteria (i.e., c and b). If $\max_i(v_{i,j}) = 0$ applies, Equation 5 results in an illegal operation if the criterion is a benefit. Similarly, if the criterion is a cost and $v_{a,j} = 0$, Equation 5 also results in an illegal operation. This issue can be circumvented by using a small value greater than 0 instead of 0. The second problem does not stem from a mathematical perspective but rather from the application of this method in a fully automated IRS. Since the SAW method only considers criteria $\mathcal{CR}$ from the applicable response set $\mathcal{R}$, it does not take into account the impact I of an intrusion. As a result of this limitation, it is possible that a response incurring high costs may be chosen even for a minor intrusion. Although this is a significant challenge for the application of SAW in IRSs, this drawback has not been addressed in existing research.

To tackle this problem, it is mandatory to set the preference value p (see Equation 6) into relation with the intrusion impact I. For each asset A of the vehicle reference architecture and each intrusion result $\mathcal{R}$, a normalized intrusion impact can be calculated. Such a normalized intrusion impact must be calculated for each metric S, F, O, P and E of the adapted HEAVENS method in Equation 2. This behavior is formulated in Equation 8.

$$\alpha_{\{S,F,O,P,E\},A,\mathcal{R}} =$$

$$\begin{cases} \frac{w_{\{S,F,O,P,E\},A,\mathcal{R}} \cdot v_{\{S,F,O,P,E\},A,\mathcal{R}}}{\sum_{|\mathcal{R}|}(w_{\{S,F,O,P,E\},A} \cdot v_{\{S,F,O,P,E\},A})}, & \text{if } \sum_{|\mathcal{R}|}(w_{\{S,F,O,P,E\},A} \cdot v_{\{S,F,O,P,E\},A}) \neq 0 \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

Similar to Equation 6, a weighted sum must be calculated. But, since the individual weights w are already included in Equation 8, a simple summation over all metrics S, F, O, P and E of the adapted HEAVENS method is sufficient. This sum will be set into relation with the preference value of the responses from Equation 6, such that the response r_i with the highest preference value p will be used, which is below the sum of all normalized HEAVENS values as depicted in Equation 9.

$$\text{best response} = \max\left\{ p_i \mid p_i < \rho \cdot \sum_{l \in \{S,F,O,P,E\}} \alpha_{l,A,\mathcal{R}} \right\} \tag{9}$$

The parameter ρ in Equation 9 is a parameter to adjust larger deviations in the order of magnitude between the sum of the normalized HEAVENS and the preference value p.

4.3.2. Adopting of Linear Programming

The first step to adopt the LP is defining the objective function. For the set of possible responses $\mathcal{R}$, it is possible to define two different objective functions:

- The first option of an objective function follows the principle of maximum benefit as depicted in Equation 10. The goal is to solve the binary decision vector $\overrightarrow{s}$ to maximize the benefit b. Although this can lead to very good solutions, it is possible that the best executable response is not found immediately since preconditions of identified responses are not satisfied.

$$\sum_{i=1}^{|\mathcal{R}|} s_i b_i \rightarrow \max \tag{10}$$

- The second option of an objective function follows the minimum cost principle and is comparable to existing IRSs [29, 27]. Equation 11 therefore leads to more conservative responses since the cost c will be minimized and the benefit b of a response is not considered. A drawback is that the identified solution inside $\overrightarrow{s}$ might not heal the system completely and another try might be necessary.

$$\sum_{i=1}^{|\mathcal{R}|} s_i c_i \rightarrow \min \tag{11}$$

For both objective functions from Equation 10 and 11 the same constraints must be satisfied for a response to qualify for execution. Existing constraints of IRSs using LP [29, 27] are not suitable for an automotive IRS. Because of that, specific constraints must be elaborated:

1. The cost c of the response must be below the impact I of the detected intrusion [29]. Equation 12 depicts this first constraint.
$$\sum_{i=1}^{|\mathcal{R}|} s_i c_i < I \tag{12}$$
2. Only one response can and must be executed as depicted in Equation 13.
$$\sum_{i=1}^{|\mathcal{R}|} s_i = 1 \tag{13}$$

It is additionally necessary that $\overrightarrow{s}$ is a binary vector, leading to the variable definition $s_i \in \{0, 1\}$.

5. Proposed Automotive IRS

In this section, we will discuss some design decisions regarding our proposed automotive IRS (refer to § 5.1) and detail its components (refer to § 5.2).

5.1. IRS Deployment

Our proposed automotive IRS can be deployed in three different locations:

- Central Gateway: The vehicle will have one IRS that receives information from various ECUs. This central IRS will have a comprehensive view and understanding of the entire system. However, it is considered a single point of failure.
- Domain Gateway: The vehicle will have one IRS per domain gateway. Each one will be mainly responsible for the ECUs belonging to that domain and will interact with other IRSs. Implementing this solution requires the existence of an Intrusion Response eXchange Protocol (IRXP) [24].
- ECU: The vehicle will have one IRS per ECU. This IRS will be primarily responsible for reacting to attacks related to its host ECU. Simultaneously, it can exchange responses related to other ECUs if needed. Choosing this option ensures the absence of a single point of failure. However, deploying such a solution requires that each ECU is capable of running the IRS, and it also necessitates the existence and the support of an IRXP [24].

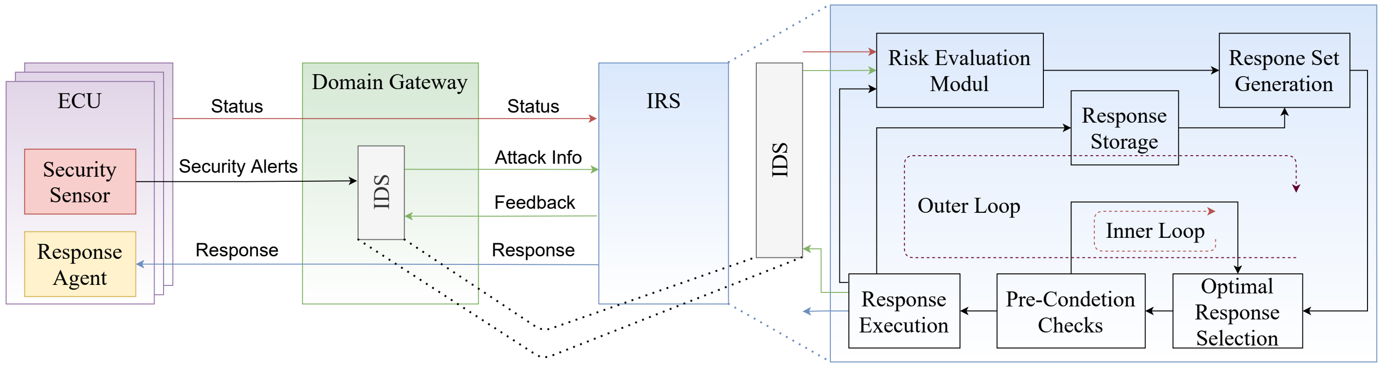


Figure 4: Internal architecture of the proposed IRS

The architecture depicted in Figure 4 illustrates the scenario where the IRS is deployed in the central gateway. Any potential change would be primarily associated with the source of certain information required for the functionality of the IRS, whether it originates from the same ECU (in the case of implementing the IRS per ECU) or from external sources such as other ECUs or domains at the gateway. Regardless of the chosen deployment location for the IRS, it necessitates the reception and sharing of information with other components within the vehicle, as outlined below:

- Attack Information: This information is provided by the IDS. In our research, we consider the IDS functionality as trusted, treating it as a black-box that reliably detects intrusions without requiring additional false-positive handling [28, 64]. In our architecture, we place the IDS in the domain gateway. Consequently, a security sensor [3] is needed to monitor its portion of the environment for security-related observations. This data is then reported to the domain-specific gateway, which houses the domain IDS.

- Status Information: This includes information about the various states of the vehicle and its surroundings. This data is collected and aggregated from various vehicle sensors and shared with the IRS.

- Response Information: This information can encompass the precise responses needed for specific ECUs or those that need to be shared with the SOC. In our architecture, we assume the presence of response agents located in each ECU. These agents are responsible for receiving responses and deploying them within the respective ECU.

It is crucial to mention the necessity of ensuring the security of this data by implementing secure communication between the ECU, domain gateway, and the IRS.

5.2. IRS component

The IRS consists of the following sub-components (as shown in Figure 4):

- Risk Evaluation Module: This module will be responsible for assessing the impact of an intrusion. The component will receive information about the intrusion from the IDS as well as information about the vehicle status.

- Response Set Generation: This module compiles a list of possible responses, utilizing information obtained from both the IDS and the risk evaluation module. Please note that not every response is applicable to every type of intrusion result (refer to Table 1).

- Optimal Response Selection: This component integrates data from all previous modules to determine the optimal response that can be applied. Within this component, any of the algorithms presented in § 4.1 can be integrated.

- Precondition Checking: Given the limitations imposed by the system architecture, where not all types of responses can be applied (for example, in cases where a sensor is unavailable due to a DoS attack, it may not always be possible to use a redundant source of information from another sensor if such a backup sensor does not exist), it is imperative to verify whether the selected optimal response is applicable or if an alternative response must be chosen. The Precondition Checking module receives the chosen response and assesses its feasibility. If a response is found to be inapplicable, a feedback loop is established with the previous Optimal Selection Module. This *inner loop* is repeated until the necessary preconditions for an individual response are met. The order of the Optimal Response Selection and the Precondition Checking is carefully evaluated and results in time benefits:

  1. ”Check-First-Then-Select”: The logical order of first eliminating all inapplicable responses and subsequently selecting the best response r from the remaining available options is illustrated by the timing behavior of Equation 14.

$$t = \left(\sum_{i=1}^{|\mathcal{R}|} t_{check,r_i} \right) + t_{select,r} + t_{execute,r} \qquad (14)$$

  The time to select the optimal response $t_{select,r}$ and the time to execute the response $t_{execute,r}$ are summed only once, since the selected response will satisfy the preconditions. In contrast, the time to check the preconditions $t_{check,r}$ is summed over the set of possible responses $\mathcal{R}$, since every response's precondition will be checked.

2. "Select-First-Then-Check": While a response may be applied with the probability p, it might also be that the constraints are not satisfied with a probability $(1 - p)$. This leads to a timing behavior of Equation 15.

$$t = t_{select,r_1} + t_{check,r_1} + p \cdot t_{execute,r_1} + (1 - p) \cdot \sum_{i=2}^{|\mathcal{R}|} \left(t_{select,r_i} + t_{check,r_i}\right) \tag{15}$$

While the first selected response must always be checked, it is only executed with the probability p. If the preconditions are not satisfied, the *Inner Loop* will be repeated maximum $|\mathcal{R}| - 1$ times.

It is evident that for a certain number of responses approaching infinity, Equations 14 and 15 yield the same runtime t when $p = 0.5$. For higher values of p, the runtime as per Equation 15 is even lower. This holds true even when $t_{select,r}$ decreases, as the number of possible responses decreases accordingly. Based on these equations, the architecture depicted in Figure 4 exhibits a "Select-First-Then-Check" behavior.

- Response Execution: This component is responsible for transmitting the chosen response initially to the domain-specific gateways and subsequently to the respective ECUs for implementation through their local response engines. After a predefined duration, this component triggers the IDS to assess the effectiveness of the applied response in mitigating the intrusion. By incorporating this IDS-Feedback loop, the *Outer Loop* can be iterated multiple times, each iteration involving a system re-evaluation. This concept serves to counter persistent attacks or stepping-stone attacks effectively. Furthermore, the feedback loop can be utilized to update the parameters of the risk evaluation module for addressing future intrusions.

An essential consideration in the IRS architecture shown in Figure 4 is the implementation of termination criteria for the *inner* and *outer loop*. The absence of such criteria could lead to an endless loop, posing a risk to the stability of the entire IRS system. While some prior research has addressed termination criteria [24, 60], these methods often involve complex evaluation techniques [10, 32] or rely on artificial intelligence support [44]. However, the high computational requirements and intricate modeling approaches associated with these methods are impractical for automotive infrastructure. To address the challenge of preventing endless loops in both the *inner* and *outer* loops, we employ two distinct methods.

1. Preventing Inner Endless Loops: To avoid an endless evaluation of preconditions, we continuously reduce the possible response set by eliminating non-applicable responses. Additionally, we've introduced a special response, labeled as "No Action" (indexed as 31), which will consistently lead to the last possible response. This specific response carries the highest cost, similar to the impact of an intrusion, but provides no benefit. These attributes ensure that the *inner loop* never reaches a deadlock since "No Action" can always be applied.
2. Avoiding Outer Endless Loops: Once a response is applied, the system undergoes an analysis through the IDS-Feedback mechanism to identify if a new stepping-stone attack is detected or if the system is secure. In case a new stepping-stone attack is detected, the entire *outer loop* illustrated in Figure 4 reiterates. To prevent an endless loop scenario when the same response is repeatedly applied, we implement changes to the parameters of the applied response based on the success of the response. The parameter adaptation differs between a successful and a non-successful response. When the selected response is unsuccessful, it indicates that the benefit values assigned to all HEAVENS parameters may not be accurate. Consequently, an adjustment is needed, resulting in a reduction of the benefit values for all HEAVENS parameters in the previously applied response. This entails the assumption that the relative order of each parameter remains unchanged; for example, if the safety benefit held a higher value than the financial benefit prior to the adjustment, it will continue to do so afterward. This behavior is mathematically expressed in Equation 16.

$$\forall i \in \{S, F, O, P\} : \quad i_{\text{new}}(i_{\text{old}}) = \begin{cases} 10, & \text{if } i_{\text{old}} = 100 \\ 1, & \text{if } i_{\text{old}} = 10 \\ 0, & \text{if } i_{\text{old}} = 1 \text{ or } i_{\text{old}} = 0 \end{cases} \tag{16}$$

A similar parameter adaptation is required in case the response was applied successfully. However, the parameters cannot simply be increased, as this could lead to predictable responses. Predictable responses pose security risks, as attackers can exploit this behavior [9]. For that reason, two adaptations are made if the response is successful to avoid predictable behavior:

- Original values are restored if the response was previously not successful and its values were adapted according to Equation 16.
- In a second step, the corresponding weights $w_{i \in S,F,O,P}$ are randomly adjusted using a prefactor r, where $r_{min} \leq r \leq r_{max}$. This retains the original order of magnitude of w_i while introducing sufficient variation through the multiplication $r \cdot w_i$ to generate different results in the next iteration.

Note that Equation 16 presented earlier does not account for the dynamic environmental parameter, denoted as E, and its corresponding weight, w_E. Further details and definitions are necessary to incorporate this parameter into the adaptation process. These details should encompass various aspects of the vehicle's status and its surrounding environment. For simplicity, we have focused on the vehicle's velocity as a parameter that can help represent the vehicle's status. To determine a realistic rating for the impact of vehicle speed, several factors must be taken into account. Studies of traffic accidents have revealed that the impact is influenced not only by the types of vehicles involved but also by their positions at the potential crash site [36]. Additionally, the age of the passengers in the vehicles can affect the impact of injuries in a traffic accident [56]. Based on this research, the approach presented in Equation 17 is applied to the parameter E in the adapted HEAVENS method's prototype implementation [36, 56].

$$E(v) = \begin{cases} 100, & \text{if } v \geq 75\ km/h \\ 10, & \text{if } 50\ km/h \leq v < 75\ km/h \\ 1, & \text{if } 30\ km/h \leq v < 50\ km/h \\ 0, & \text{if } 0\ km/h \leq v < 30\ km/h \end{cases} \tag{17}$$

- Response Storage: Within this component, a repository is maintained containing a range of potential responses alongside their associated metrics. These metrics can be updated through the feedback mechanism or expanded with the inclusion of new responses and parameters via an external connectivity interface. When implementing this on specific hardware, it is crucial to implement security measures to prevent unauthorized tampering with the memory area.

Our proposed IRS architecture, featuring both an *inner loop* and an *outer loop*, coupled with the incorporation of automotive-specific considerations into the external architecture, introduces a novel paradigm in the realm of fully automated IRSs. Note that there is already some related work for each part of the IRS (such as the selection method), which was covered in the previous sections. However, there is no system that attempts to include all the aspects against which we can compare our work.

6. Evaluation

6.1. Implementation, Testbed, and Use Cases

The proposed IRS was implemented using the Python programming language. To implement Linear Programming and the associated Simplex algorithm, we utilized the `PuLP library` [49], a well-established choice, along with the GNU Linear Programming Kit as the solver. It's important to note that the adapted SAW method remains independent of this decision, as it relies solely on standard Python mathematical operators.

The testbed designed for evaluating the Intrusion Response Engine (IRE) incorporates an embedded system setup to realistically emulate the automotive infrastructure. To ensure this fidelity, our implementation was executed on a Raspberry Pi 4 Model B Rev 1.2, a choice justified by the device's ARM-based quad-core processor running at 1.5 GHz. This processing power closely aligns with the high-performance chips commonly found in the automotive industry.

The goal of the evaluation is to assess two key aspects of the proposed IRS. Firstly, we aim to evaluate its proficiency in optimal response selection, and secondly, we intend to measure various computational metrics, including memory consumption and the time required to obtain optimal responses while using the three different selection algorithms: LP with maximum benefit, LP with minimum cost, and adapted SAW.

For our evaluation, we employed two representative intrusion scenarios inspired by real-world intrusions:

1. Adversarial Sample: This scenario involves slight modifications to the input data of a machine learning algorithm, resulting in significantly different outputs from the original [46]. Given the prevalent use of machine learning algorithms in cameras for automated vehicles, they are vulnerable to exploitation via adversarial samples [46]. In our evaluation, we exploited a front camera in a rural setting, leading to an altered behavior in the acceleration control.
2. Information Disclosure at the Infotainment System: This scenario draws inspiration from an actual attack on a vehicle, where an information disclosure in the infotainment system served as the initial step in a stepping-stone attack [48].

The specific IDS parameters and vehicle states employed as input for the scenarios are meticulously detailed in Table 3. Please remember that in our prototype of the IRS, we consider only the velocity of the attacked vehicle as an illustrative example of a vehicle's status.

6.2. Result

In this section, we will present the results of testing our IRS using two prominent scenarios. We will evaluate response quality, response selection time, memory consumption, and the adaptation of response parameters for each of the three selection algorithms: LP with maximum benefit, LP with minimum cost, and the adapted SAW.

6.2.1. Response Quality

The objective of the response quality evaluation is to assess how different optimal selection algorithms prioritize responses and determine the overall impact and benefit of the applied responses. To achieve that, the precondition of each response is set to 'rejected' for every proposed response. This ensures that the IRS will continue to suggest responses from the list of possible responses. Each applied response can have both positive and negative effects on the system, so the cost and benefit values of the selected responses are presented. In this evaluation, default parameters are utilized for each new test, ensuring uniformity in the algorithm evaluation across various metrics.

Table 3: IDS-related information and vehicle state parameters for both evaluation scenarios.

Property	Scenario 1	Scenario 2
Name	Adversarial sample	Information disclosure at the infotainment system
Infected Asset	Front Camera	Infotainment Gateway
Affected Asset	Acceleration control	Infotainment Gateway
Intrusion Result	Falsify / Alter behavior	Information Disclosure
Dynamic Parameter	Velocity: 70 km/h	Velocity: 0 km/h

Figures 5 depicts the cost and benefit of all proposed responses in the order they are applied by the respective algorithm for the both scenarios. The figure shows that our proposed IRS suggests a different number and order of responses for various scenarios and for different selection algorithms within the same scenario. Please note that the figure shows that some responses were selected twice. For example, the response of restarting the misbehaving system (indexed with number 19, see Table 1), was selected twice. However, it's important to clarify that the response was selected for different systems. In other words, the first restart is related to the camera, while the second is for the acceleration control. In addition, as expected and shown in Figure 5a and Figure 5b, the LP method with maximum benefit starts at a very high benefits. Similarly, the LP with minimum response costs starts at a very low cost and more expensive responses are not selected until later stages, as shown in Figure 5c and Figure 5d. Notably, when the LP with maximum benefit works arbitrarily with respect to the cost, but the constraint that the cost of the response is below the impact of the intrusion is always fulfilled (see Equation 12). The reason for the arbitrary behavior is that Linear Programming only follows one optimization function and just satisfies the constraints, but does not sort by constraints. Similarly, LP with minimum cost delivers arbitrary values with respect to the benefit because it only considers cost metrics in its optimization. While the LP with the minimum cost provides more conservative solutions, the LP with maximum benefit suggests more offensive solutions. In a real-world scenario, LP with minimum cost might require multiple responses since its benefits are arbitrarily sorted, while LP with maximum benefit might require more iterations of the "inner loop" since the preconditions for more offensive responses might not be fulfilled.

The adapted SAW method exhibits a similar arbitrary behavior as shown in Figure 5e and Figure 5f. However, it is noticeable that adapted SAW may select responses with a cost higher than the impact of the intrusion (see Figure 5f). Given that the adapted SAW method does not consider constraints, it is an unattractive solution to use any SAW method in an automatic IRS.

6.2.2. *Time of Response Selection*

To evaluate the time required for selecting a response from a given response list using the selection algorithms, we utilized the previously described method where the *inner loop* of the IRS repeats multiple times. It's important to note that the generation of the response set occurs only once for an individual intrusion. The time required for list generation is independent of the selection algorithm, measuring at 4.32 ms for scenario 1 and 3.82 ms for scenario 2. The difference in the measured time between the scenarios is due to the variation in number of possible responses.

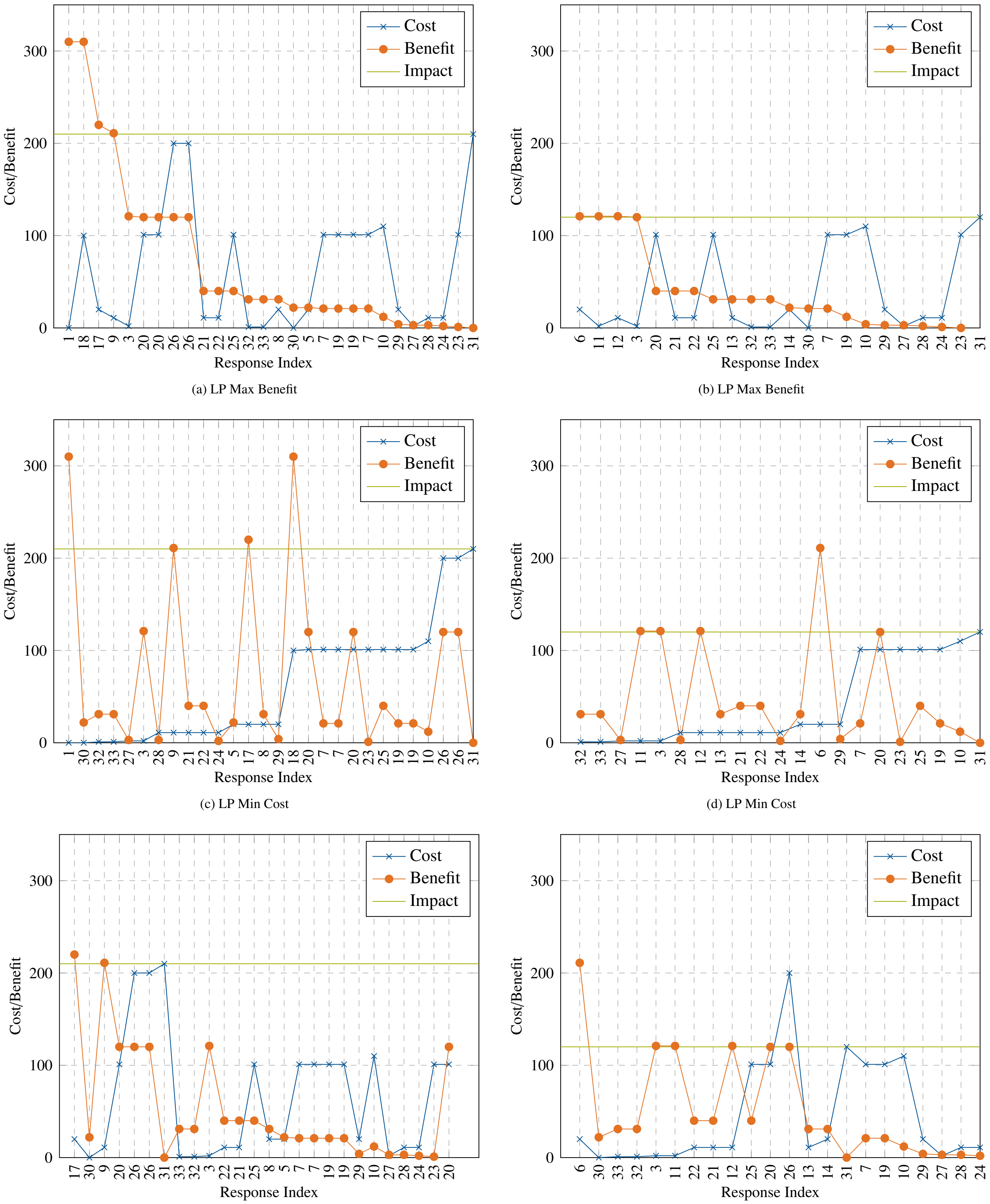
Table 4: Memory consumption of the IRS in kB using static evaluation.

	LP with Max Benefit	LP with Min Cost	Adapted SAW
Scenario 1	19308	19206	11296
Scenario 2	19228	19344	11220

Figure 6 illustrates the time consumed by the three selection algorithms during the process of selecting different responses. Please note that the X-axis represents the order of the response, not the index of the response. The figure indicates that the adapted SAW method consumes less time compared to the LP methods. Specifically, the LP method with maximum benefit typically consumes more time due to the need for multiple iterations, as its offensive responses may not meet necessary preconditions. Slightly less time is needed for the LP method with minimum cost, although its conservative responses are selected after fewer precondition checks. Overall, all algorithms demonstrate good performance on a resource-constrained embedded system.

6.2.3. *Memory Consumption*

To measure memory consumption, we utilized Python's internal `resource` module [55]. Since some of the optimal selection algorithms rely on third-party libraries, the assessment of memory consumption includes the memory allocated for these functionalities as well. The results are presented in Table 4. The results show that both LP with maximum benefit and LP with minimum cost methods consume nearly the same amount of memory, while the adapted SAW method exhibits considerably lower memory consumption. This difference can be attributed to the external libraries `PuLP` and the `GNU Linear Programming Kit`, which require more memory due to their complex data structures and solving methods. Nevertheless, all three selection algorithms exhibit low memory consumption, making them suitable for use in resource-constrained embedded hardware systems.



(a) LP Max Benefit

(b) LP Max Benefit

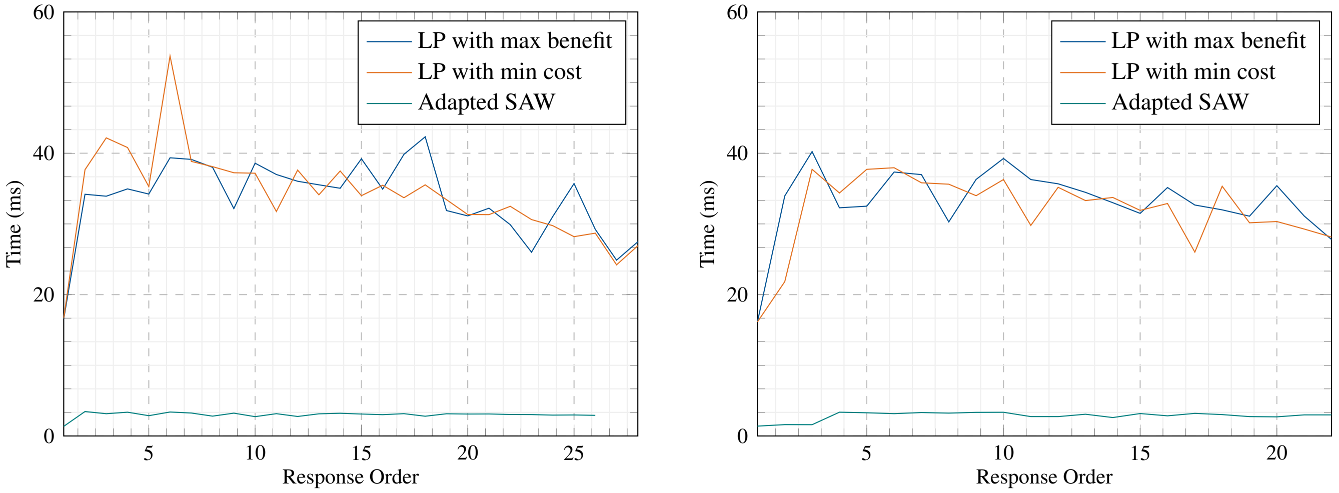
(c) LP Min Cost

(d) LP Min Cost

(e) SAW

(f) SAW

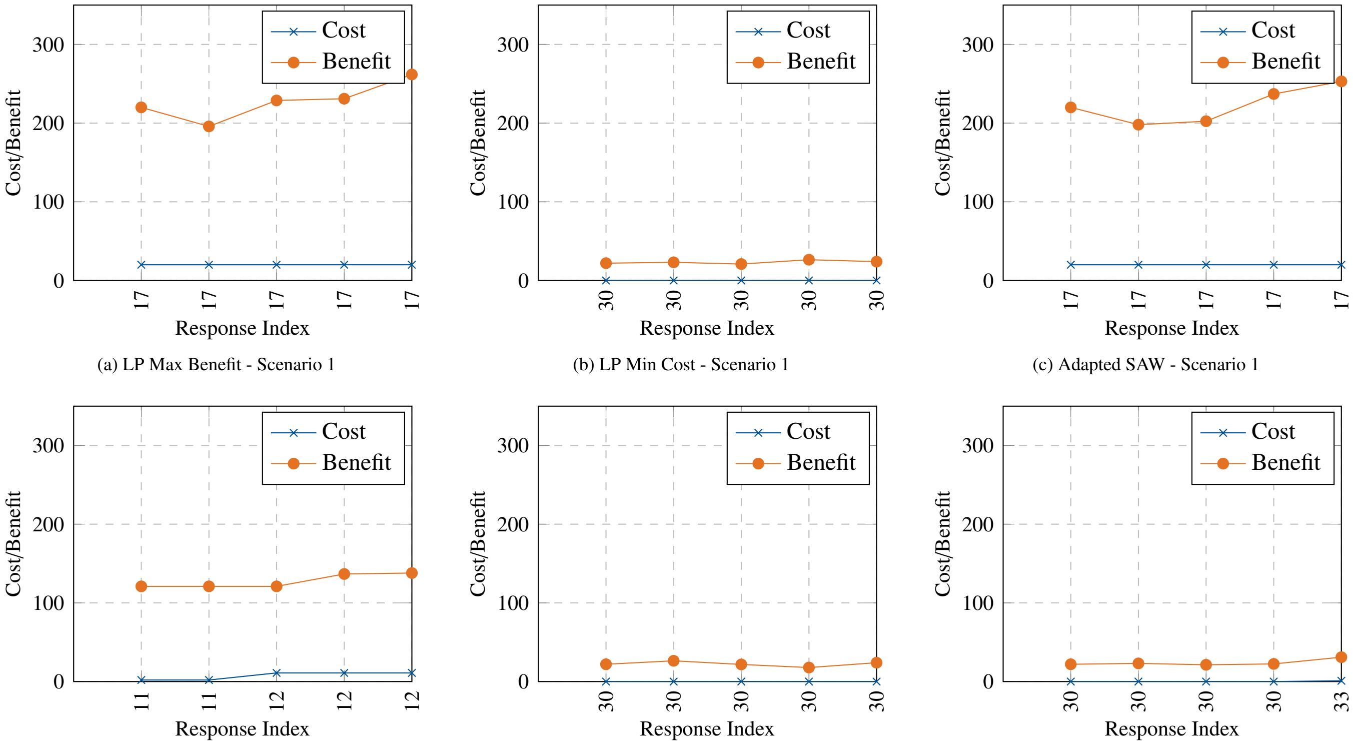
Figure 5: Evaluation of the response benefit and cost for Scenario 1 (left) and Scenario 2 (right) using LP with maximum benefit (top), LP with minimum cost (middle), and adapted SAW (bottom).



(a) Scenario 1

(b) Scenario 2

Figure 6: Evaluation of consumed time for response selection using the three selection algorithms for both scenarios.



(a) LP Max Benefit - Scenario 1

(b) LP Min Cost - Scenario 1

(c) Adapted SAW - Scenario 1

(d) LP Max Benefit - Scenario 2

(e) LP Min Cost - Scenario 2

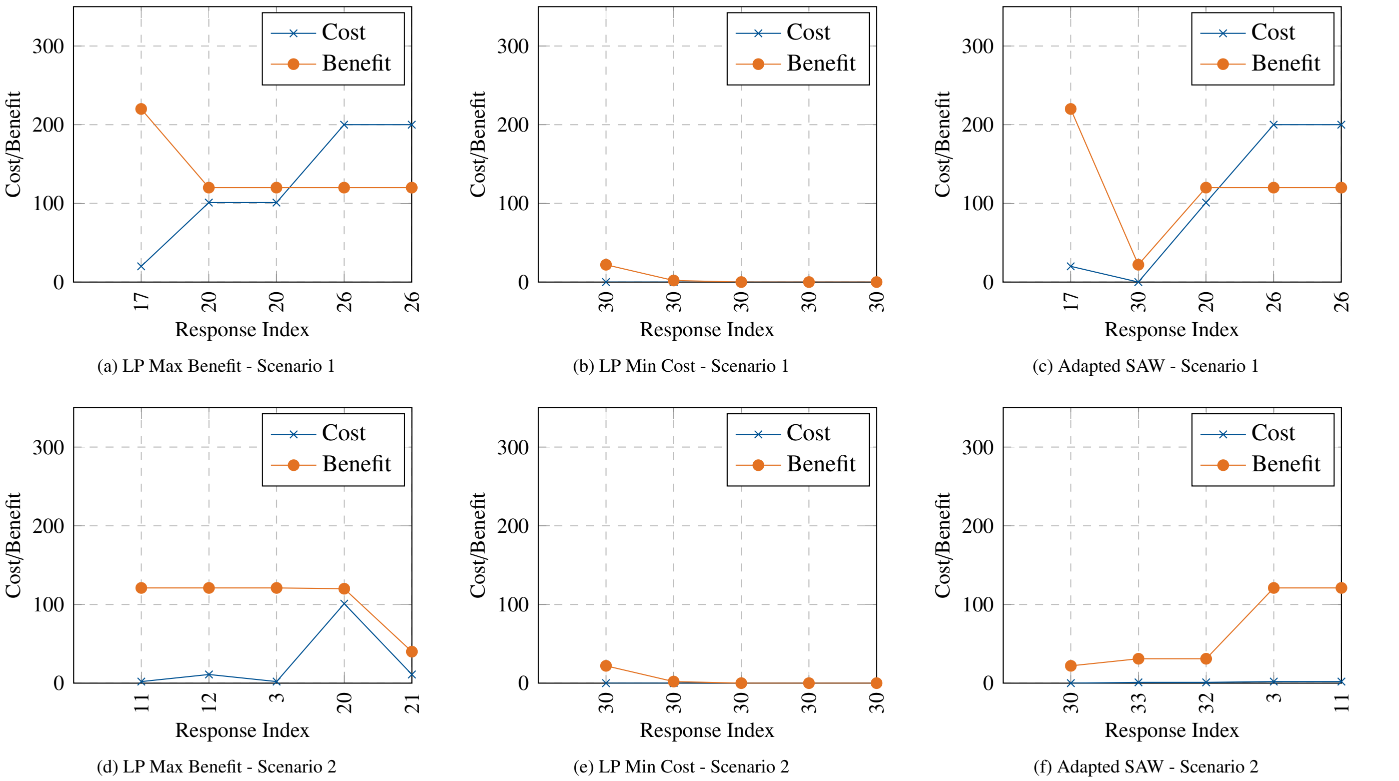
(f) Adapted SAW - Scenario 2

Figure 7: Evaluation of parameter adaptation in Scenario 1 (top) and Scenario 2 (bottom) for the responses selected over five iterations using the three selection algorithms, assuming the responses were consistently considered successful.

6.2.4. Dynamic Evaluation

The dynamic evaluation concentrates on two key aspects: response and threat impact parameters adaptation (refer to § 3) and the inclusion of velocity considerations (as shown in Equation 17). When it comes to parameters adaptation, response quality is assessed based on their cost and benefit. In terms of velocity, we evaluate response variation. These assessments are conducted for both scenarios 1 and 2. By testing all three implemented optimal selection algorithms, we can compare their dynamic behavior.

Parameters adaption. To assess the impact of changing parameters, we conducted two repetitions of each scenario, each comprising five iterations of the *outer loop*. In one set of iterations for each scenario, we consistently deemed the responses as successful, while in the other set of five iterations, the responses were uniformly considered unsuccessful. The benefits and costs of the five optimally selected responses for both sce-



(a) LP Max Benefit - Scenario 1

(b) LP Min Cost - Scenario 1

(c) Adapted SAW - Scenario 1

(d) LP Max Benefit - Scenario 2

(e) LP Min Cost - Scenario 2

(f) Adapted SAW - Scenario 2

Figure 8: Evaluation of parameter adaptation in Scenario 1 (top) and Scenario 2 (bottom) for the responses selected over five iterations using the three selection algorithms, assuming the responses were consistently considered unsuccessful.

narios, as determined by the three selection algorithms, under the assumption that the responses were always successful, are presented in Figure 7. Correspondingly, the results under the assumption that the responses were consistently unsuccessful are displayed in Figure 8.

In consistently successful attacks, we observed that parameter weights change within the range of ±20% (we have selected $r_{min} = 0.8$ and $r_{max} = 1.2$). The purpose of these changes was to reduce response predictability. In both scenarios, changes in response benefit were evident. However, in the first scenario, all three algorithms retained the same response as shown in Figure 7a, 7b, and 7c. This was changed in the second scenario, where responses were altered for the LP with maximum benefit and adaptive SAW algorithms as shown in Figure 7d, and 7f. The reason for the absence of changes in the selected responses in the first scenario when using LP with maximum benefits or adapted SAW algorithms can be attributed to the specific response chosen: transitioning to a safe mode (indexed with 17). This response had very high benefit values, as determined through the initial evaluation process, making minor variations of ±20% inconsequential to the overall result. Consequently, minor variations of ±20% did not affect the overall result, as the next possible response had significantly lower benefit values. To avoid such a constant behavior, a more substantial modification of the response parameters or the use of an asymmetric window for the prefactor, with a higher probability of negative values, can be implemented. Notably, the LP method with minimum cost (Figure 7b and 7e) did not consider response benefits in its optimization function, rendering modifications to response benefit irrelevant. This method-related limitation persisted across both simulated scenarios.

In the case of consistently unsuccessful attacks, we observe more substantial variations in the selected responses compared to the previous case (see Figure 8). This behavior is expected, as the parameter adaptation in a non-successful case involves higher orders of magnitude, as shown in Equation 16, compared to the successful case. Similar to the previous analysis, the LP method with minimum cost optimization consistently generates the same response due to the exclusion of response benefit in the optimization process, as shown in Figures 8b and 8e. Conversely, LP with maximum benefit optimization aligns with expectations. Although the initial response is similar to the successful case, subsequent responses exhibit lower benefits (Figures 8a and 8d) and higher costs as a side effect. Notably, response index 26 (killing the process) appeared twice in Figures 8a and 8c, each referring to different components (i.e., camera and acceleration control). The adapted SAW method consistently produces varying results with less distinct trends in benefit and cost when compared to LP with maximum benefit (Figures 8c and 8f). This observed behavior holds true for both scenarios1 and 2, underscoring the expected functionality of parameter adaptation for non-successful cases.

In conclusion, this assessment of dynamic parameter adaptation confirms that LP with maximum benefit and the adapted SAW methods perform effectively with adjusted parameters, rendering the results valid for both test cases. On the other

Table 5: Impact of the velocity for the evaluated scenarios, using Equation 2.

	Impact (unitless)		
	0 km/h	**50 km/h**	**100 km/h**
Scenario 1	200	210	300
Scenario 2	120	130	220

hand, the LP method with minimum cost optimization falls short in its capacity to respond to parameter shifts in response benefit values. Consequently, this method appears less appealing for identifying optimal responses in autonomous IRS.

Inclusion of Velocity Considerations. The second key aspect of dynamic evaluation involves assessing the influence of vehicle velocity on the selected responses. In our current prototype system, the environmental parameter E is treated similarly to other HEAVENS parameters in Equation 2, as their respective weights w are either one or zero. As we alter the velocity, the environmental parameter for an intrusion takes on different values, as indicated in Equation 17. Therefore, intrusion the impact is more significant at higher velocities. For this test, both scenario one and two are assessed at three velocities: 0, 50, and 100 km/h, using all three implemented algorithms, with each evaluation beginning with the default data-set.

While the intrusion impact calculation in Table 5 functions as expected, each algorithm consistently selects the same response within each scenario, regardless of the velocity. This behavior can be attributed to the high impact values in the two evaluated scenarios. In cases of less severe intrusions or during the early stages of a stepping-stone attack, where the HEAVENS parameters result in lower values, the velocity's impact becomes relatively more substantial, thus leading to varying results. Nonetheless, it's important to emphasize that the proposed IRS architecture is adaptable since the individual weights w for HEAVENS parameters can be customized as per Equation 2. This customization minimizes the over-representation of static HEAVENS parameters, enabling the velocity to exert a more pronounced influence on the selected response.

6.2.5. Final Remarks

The evaluation of the developed IRS reveals the advantages and drawbacks of each selection method. The adapted SAW method is limited by its inability to consider constraints. Consequently, it is not feasible to employ this method in a fully automated IRS. On the other hand, LP with minimum cost consistently favors constant responses and is therefore unsuitable for optimal response identification. Despite its successful application in existing research [29, 27], the results demonstrate suboptimal behavior for the automotive use case. Nevertheless, it is well-suited for proposing follow-up responses once the primary intrusion has been mitigated. These follow-up responses can enhance security by alerting a SOC and providing information to the car manufacturer, ultimately leading to updated software. In contrast, the LP method with maximum benefit, excels in all metrics evaluated for an automotive IRS. Since it offers responses with high benefits from the outset, it is well-suited to respond to the primary intrusion.

7. Conclusion and Outlook

Modern vehicles' intricate architecture and advanced connectivity present unique intrusion challenges. While automotive security research has traditionally emphasized IDSs as a secondary defense layer, the development of vehicle IRS is in its early stages, drawing inspiration from related industries. To delve into the development of an automotive IRS, we sought answers to three key questions: defining potential responses, outlining response evaluation criteria, and optimizing response selection. Initially, we categorized automotive intrusions and stepping-stone attacks into five distinct categories to create a more versatile intrusion model. Similarly, we classified responses, creating a formal description for both intrusions and responses. Additionally, we investigated necessary adjustments to existing risk assessment models to support response evaluation. Furthermore, we conducted a comprehensive comparison of various optimal selection algorithms, highlighting the adaptability of the SAW method and Linear Programming (LP) with various optimizations for IRS integration. Although other algorithm families may gain relevance in the future, they currently face limitations in the automotive context. In addition to these findings, we proposed an IRS architecture that accommodates the distributed nature of vehicles and addresses automotive-specific constraints. Evaluation in real-world scenarios has led to the development of a novel vehicular IRS, demonstrating its potential for integration into modern distributed vehicle architectures and enhancing overall security.

While the focus of the paper is on the analysis and design of the IRS, the implementation of the external architecture and the response execution modules on the local engines on each ECU is still a challenge towards an IRS as a system. To test such an overall IRS system, real-world data-sets including both normal operation and the attack scenarios are needed. Extensive evaluation in Software-in-the-Loop or Hardware-in-the-Loop testbeds can extend the existing evaluations of algorithms and the overall system. With respect to the secure communication of intrusions and responses, further research and standardization needs to be performed in order to ensure that the developed IRS does not only reply in an adequate manner, but also distributes its responses. The modular architecture of the IRS allows an easy extension towards more complex vehicle architectures and new intrusions or responses. Additionally it allows the integration of new selection algorithms in the future to adapt to possible changed needs.

Acknowledgement

This work is supported by the European Union-funded projects CyberSecDome (Agreement No.: 101120779).

References

[1] Alrefaei, F., Alzahrani, A., Song, H., Alrefaei, S., 2022. A survey on the jamming and spoofing attacks on the unmanned aerial vehicle networks, in: 2022 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS), IEEE. pp. 1–7.

[2] Anuar, N.B., Papadaki, M., Furnell, S., Clarke, N., 2012. A Response Strategy Model for Intrusion Response Systems, in: Gritzalis, D., Furnell, S., Theoharidou, M. (Eds.), Information Security and Privacy Research, Springer Berlin Heidelberg, Berlin, Heidelberg. pp. 573–578.

[3] Anwar, S., Mohamad Zain, J., Zolkipli, M.F., Inayat, Z., Khan, S., Anthony, B., Chang, V., 2017. From Intrusion Detection to an Intrusion Response System: Fundamentals, Requirements, and Future Directions. Algorithms 10. doi:10.3390/a10020039.

[4] Anwar, S., Zain, J.M., Zolkipli, M.F., Inayat, Z., Jabir, A.N., Odili, J.B., 2015. Response option for attacks detected by intrusion detection system, in: 2015 4th International Conference on Software Engineering and Computer Systems (ICSECS), pp. 195–200. doi:10.1109/ICSECS.2015.7333109.

[5] AUTOSAR, 2020. Specification of Intrusion Detection System Protocol. Technical Report. AUTOSAR Consortium. URL: https://www.autosar.org/fileadmin/standards/R20-11/FO/AUTOSAR_PRS_IntrusionDetectionSystem.pdf.

[6] Barletta, V.S., Caivano, D., Vincentiis, M.D., Ragone, A., Scalera, M., Martín, M.Á.S., 2023. V-soc4as: A vehicle-soc for improving automotive security. Algorithms 16, 112.

[7] Bashendy, M., Tantawy, A., Erradi, A., 2023a. Intrusion response systems for cyber-physical systems: A comprehensive survey. Comput. Secur. 124. doi:10.1016/j.cose.2022.102984.

[8] Bashendy, M., Tantawy, A., Erradi, A., 2023b. Intrusion response systems for cyber-physical systems: A comprehensive survey. Computers & Security 124, 102984.

[9] Bouyahia, T., Cuppens-Boulahia, N., Cuppens, F., Autrel, F., 2017. Multi-Criteria Recommender Approach for Supporting Intrusion Response System, in: Cuppens, F., Wang, L., Cuppens-Boulahia, N., Tawbi, N., Garcia-Alfaro, J. (Eds.), Foundations and Practice of Security, Springer International Publishing, Cham. pp. 51–67.

[10] Cardellini, V., Casalicchio, E., Iannucci, S., Lucantonio, M., Mittal, S., Panigrahi, D., Silvi, A., 2022. An Intrusion Response System utilizing Deep Q-Networks and System Partitions. https://arxiv.org/abs/2202.08182. doi:10.48550/ARXIV.2202.08182.

[11] Checkoway, S., McCoy, D., Kantor, B., Anderson, D., Shacham, H., Savage, S., Koscher, K., Czeskis, A., Roesner, F., Kohno, T., 2011. Comprehensive experimental analyses of automotive attack surfaces, in: 20th USENIX security symposium (USENIX Security 11).

[12] Chevalier, R., Plaquin, D., Dalton, C., Hiet, G., 2019. Survivor: A Fine-Grained Intrusion Response and Recovery Approach for Commodity Operating Systems, in: Proceedings of the 35th Annual Computer Security Applications Conference, Association for Computing Machinery, New York, NY, USA. p. 762–775. doi:10.1145/3359789.3359792.

[13] Cui, J., Liew, L.S., Sabaliauskaite, G., Zhou, F., 2019. A review on safety failures, security attacks, and available countermeasures for autonomous vehicles. Ad Hoc Networks 90, 101823.

[14] El-Rewini, Z., Sadatsharan, K., Selvaraj, D.F., Plathottam, S.J., Ranganathan, P., 2020. Cybersecurity challenges in vehicular communications. Vehicular Communications 23, 100214. doi:https://doi.org/10.1016/j.vehcom.2019.100214.

[15] European Union Agency for Cybersecurity, 2019. ENISA Good Practices for the Security of Smart Cars. Technical Report. European Union Agency for Cybersecurity (ENISA), Greece. URL: https://www.enisa.europa.eu/publications/smart-cars.

[16] Fessi, B.A., BenAbdallah, S., Hamdi, M., Boudriga, N., 2009. A new genetic algorithm approach for intrusion response system in computer networks, in: 2009 IEEE Symposium on Computers and Communications, pp. 342–347. doi:10.1109/ISCC.2009.5202379.

[17] Fishburn, P.C., 1967. Additive utilities with incomplete product sets: Application to priorities and assignments. Operations Research 15, 537–542.

[18] Guo, Y., Zhang, H., Li, Z., Li, F., Fang, L., Yin, L., Cao, J., 2020. Decision-Making for Intrusion Response: Which, Where, in What Order, and How Long?, in: ICC 2020 - 2020 IEEE International Conference on Communications (ICC), pp. 1–6. doi:10.1109/ICC40277.2020.9149083.

[19] Hamad, M., Hammadeh, Z.A., Saidi, S., Prevelakis, V., Ernst, R., 2018. Prediction of abnormal temporal behavior in real-time systems, in: Proceedings of the 33rd Annual ACM Symposium on Applied Computing, pp. 359–367.

[20] Hamad, M., Nolte, M., Prevelakis, V., 2016. Towards Comprehensive Threat Modeling for Vehicles, in: the 1st Workshop on Security and Dependability of Critical Embedded Real-Time Systems.

[21] Hamad, M., Prevelakis, V., 2020. SAVTA: A Hybrid Vehicular Threat Model: Overview and Case Study. Information 11. doi:10.3390/info11050273.

[22] Hamad, M., Steinhorst, S., 2023. Security challenges in autonomous systems design. arXiv:2312.00018.

[23] Hamad, M., Tsantekidis, M., Prevelakis, V., 2019. Red-Zone: Towards an Intrusion Response Framework for Intra-Vehicle System, in: 5th International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS). doi:10.5220/0007715201480158.

[24] Hamad, M., Tsantekidis, M., Prevelakis, V., 2021. Intrusion Response System for Vehicles: Challenges and Vision, in: Helfert, M., Klein, C., Donnellan, B., Gusikhin, O. (Eds.), Smart Cities, Green Technologies and Intelligent Transport Systems, Springer International Publishing, Cham. pp. 321–341.

[25] Heigl, M., Doerr, L., Almaini, A., Fiala, D., Schram, M., 2018. Incident Reaction Based on Intrusion Detections' Alert Analysis, in: 2018 International Conference on Applied Electronics (AE), pp. 1–6. doi:10.23919/AE.2018.8501419.

[26] Henniger, O., Ruddle, A., Seudié, H., Weyl, B., Wolf, M., Wollinger, T., 2009. Securing vehicular on-board it systems: The evita project, in: VDI/VW Automotive Security Conference, p. 41.

[27] Herold, N., 2017. Incident Handling Systems with Automated Intrusion Response. Dissertation. Technische Universität München.

[28] Herold, N., Posselt, S.A., Hanka, O., Carle, G., 2016. Anomaly detection for SOME/IP using complex event processing, in: NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium, pp. 1221–1226. doi:10.1109/NOMS.2016.7502991.

[29] Herold, N., Wachs, M., Posselt, S.A., Carle, G., 2017. An Optimal Metric-Aware Response Selection Strategy for Intrusion Response Systems, in: Cuppens, F., Wang, L., Cuppens-Boulahia, N., Tawbi, N., Garcia-Alfaro, J. (Eds.), Foundations and Practice of Security, Springer International Publishing, Cham. pp. 68–84.

[30] Hughes, K., McLaughlin, K., Sezer, S., 2020. Dynamic Countermeasure Knowledge for Intrusion Response Systems, in: 2020 31st Irish Signals and Systems Conference (ISSC), pp. 1–6. doi:10.1109/ISSC49989.2020.9180198.

[31] Iannucci, S., Barba, O.D., Cardellini, V., Banicescu, I., 2019a. A performance evaluation of deep reinforcement learning for model-based intrusion response, in: 2019 IEEE 4th International Workshops on Foundations and Applications of Self* Systems (FAS*W), pp. 158–163. doi:10.1109/FAS-W.2019.00047.

[32] Iannucci, S., Casalicchio, E., Lucantonio, M., 2021. An Intrusion Response Approach for Elastic Applications Based on Reinforcement Learning, in: 2021 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 01–10. doi:10.1109/SSCI50451.2021.9659882.

[33] Iannucci, S., Montemaggio, A., Williams, B., 2019b. Towards self-defense of non-stationary systems, in: 2019 International Conference on Computing, Networking and Communications (ICNC), pp. 250–254. doi:10.1109/ICCNC.2019.8685487.

[34] International Organization for Standardization, 2021. ISO/SAE 21434: 2021: Road Vehicles: Cybersecurity Engineering. ISO.

[35] Islam, M.M., Lautenbach, A., Sandberg, C., Olovsson, T., 2016. A risk assessment framework for automotive embedded systems, in: Proceedings of the 2nd ACM International Workshop on Cyber-Physical System Security, pp. 3–14.

[36] Jurewicz, C., Sobhani, A., Woolley, J., Dutschke, J., Corben, B., 2016. Exploration of Vehicle Impact Speed – Injury Severity Relationships for Application in Safer Road Design. Transportation Research Procedia 14, 4247–4256. URL: https://www.sciencedirect.com/science/article/pii/S2352146516304021, doi:https://doi.org/10.1016/j.trpro.2016.05.396.

[37] Karahasanovic, A., Kleberger, P., Almgren, M., 2017. Adapting Threat Modeling Methods for the Automotive Industry, in: ej tryckt.

[38] Kholidy, H.A., Erradi, A., Abdelwahed, S., Baiardi, F., 2016. A risk mit-

igation approach for autonomous cloud intrusion response system. Computing 98, 1111–1135. doi:10.1007/s00607-016-0495-8.

[39] Kim, K., Kim, J.S., Jeong, S., Park, J.H., Kim, H.K., 2021. Cybersecurity for autonomous vehicles: Review of attacks and defense. Computers & Security 103, 102150.

[40] Klee, V., Minty, G.J., 1972. How good is the simplex algorithm?, in: Inequalities III (Proc. Third Sympos., Univ. California, Los Angeles, Calif., 1969; dedicated to the memory of Theodore S. Motzkin), Academic Press, New York. p. 159–175.

[41] Konak, A., Coit, D.W., Smith, A.E., 2006. Multi-objective optimization using genetic algorithms: A tutorial. Reliability engineering & system safety 91, 992–1007.

[42] Lautenbach, A., Almgren, M., Olovsson, T., 2021. Proposing heavens 2.0–an automotive risk assessment model, in: Proceedings of the 5th ACM Computer Science in Cars Symposium, pp. 1–12.

[43] Lokman, S.F., Othman, A.T., Abu-Bakar, M.H., 2019. Intrusion detection system for automotive Controller Area Network (CAN) bus system: a review. EURASIP Journal on Wireless Communications and Networking 2019, 184. doi:10.1186/s13638-019-1484-3.

[44] Lopes, A., Hutchison, A., 2020. Experimenting with Machine Learning in Automated Intrusion Response, in: Kotenko, I., Badica, C., Desnitsky, V., El Baz, D., Ivanovic, M. (Eds.), Intelligent Distributed Computing XIII, Springer International Publishing, Cham. pp. 505–514.

[45] Luo, F., Jiang, Y., Zhang, Z., Ren, Y., Hou, S., 2021. Threat Analysis and Risk Assessment for Connected Vehicles: A Survey. Security and Communication Networks 2021. doi:10.1155/2021/1263820.

[46] Mahima, K.T.Y., Ayoob, M., Poravi, G., 2021. Adversarial Attacks and Defense Technologies on Autonomous Vehicles: A Review. Applied Computer Systems 26, 96–106. doi:doi:10.2478/acss-2021-0012.

[47] Miller, C., Valasek, C., 2015a. Remote exploitation of an unaltered passenger vehicle. Black Hat USA 2015, 1–91.

[48] Miller, C., Valasek, C., 2015b. Remote Exploitation of an Unaltered Passenger Vehicle. https://illmatics.com/Remote%20Car%20Hacking.pdf. Accessed: 12.04.2022.

[49] Mitchell, S., O'Sullivan, M., Dunning, I., 2011. PuLP: A Linear Programming Toolkit for Python. Department of Engineering Science, The University of Auckland, Auckland, New Zealand .

[50] Nespoli, P., Papamartzivanos, D., Gómez Mármol, F., Kambourakis, G., 2018. Optimal countermeasures selection against cyber attacks: A comprehensive survey on reaction frameworks. IEEE Communications Surveys & Tutorials 20, 1361–1396. doi:10.1109/COMST.2017.2781126.

[51] Olt, C., 2019. Establishing security operation centers for connected cars. ATZelectronics worldwide 14, 40–43.

[52] Ossenbühl, S., Steinberger, J., Baier, H., 2015. Towards Automated Incident Handling: How to Select an Appropriate Response against a Network-Based Attack?, in: 2015 Ninth International Conference on IT Security Incident Management & IT Forensics, pp. 51–67. doi:10.1109/IMF.2015.13.

[53] Palanca, A., Evenchick, E., Maggi, F., Zanero, S., 2017. A stealth, selective, link-layer denial-of-service attack against automotive networks, in: Detection of Intrusions and Malware, and Vulnerability Assessment: 14th International Conference, DIMVA 2017, Bonn, Germany, July 6-7, 2017, Proceedings 14, Springer. pp. 185–206.

[54] Papadaki, M., Furnell, S., Lines, B., Reynolds, P., 2003. Operational Characteristics of an Automated Intrusion Response System, in: Lioy, A., Mazzocchi, D. (Eds.), Communications and Multimedia Security. Advanced Techniques for Network and Data Protection, Springer Berlin Heidelberg, Berlin, Heidelberg. pp. 65–75.

[55] Python Software Foundation, 2022. resource — Resource usage information. https://docs.python.org/3/library/resource.html. Accessed: 20.07.2022.

[56] Richards, D.C., 2010. Relationship between speed and risk of fatal injury: pedestrians and car occupants, in: Road Safety Web Publication, Department for Transport, London.

[57] Rose, J.R., Swann, M., Grammatikakis, K.P., Koufos, I., Bendiab, G., Shiaeles, S., Kolokotronis, N., 2022. Ideres: Intrusion detection and response system using machine learning and attack graphs. Journal of Systems Architecture 131, 102722.

[58] Schrijver, A., 1998. The simplex method, in: Theory of Linear and Integer Programming, John Wiley & Sons, New York. pp. 129–150.

[59] Sembera, V., 2020. Iso/sae 21434: Setting the standard for connected cars' cybersecurity. Trend Micro Research, White Paper .

[60] Shameli-Sendi, A., Ezzati-Jivan, N., Jabbarifar, M., Dagenais, M., 2012. Intrusion Response Systems: Survey and Taxonomy. International Journal Computer Science Network Security (IJCSNS) 12.

[61] Souissi, S., Serhrouchni, Sliman, L., Charroux, B., 2017. Security Incident Response: Towards a Novel Decision-Making System, in: Madureira, A.M., Abraham, A., Gamboa, D., Novais, P. (Eds.), Intelligent Systems Design and Applications, Springer International Publishing.

[62] Stakhanova, N., Strasburg, C., Basu, S., Wong, J.S., 2012. Towards cost-sensitive assessment of intrusion response selection. Journal of computer security 20, 169–198.

[63] Strasburg, C., Stakhanova, N., Basu, S., Wong, J.S., 2009. A framework for cost sensitive assessment of intrusion response selection, in: 2009 33rd Annual IEEE international computer software and applications conference, IEEE. pp. 355–360.

[64] Ullah, S., Khan, M.A., Ahmad, J., Jamal, S.S., e Huma, Z., Hassan, M.T., Pitropakis, N., Arshad, Buchanan, W.J., 2022. HDL-IDS: A Hybrid Deep Learning Architecture for Intrusion Detection in the Internet of Vehicles. Sensors 22. doi:10.3390/s22041340.

[65] Ullah, S., Shelly, S., Hassanzadeh, A., Nayak, A., Hasan, K., 2020. On the Effectiveness of Intrusion Response Systems against Persistent Threats, in: 2020 International Conference on Computing, Networking and Communications (ICNC), pp. 415–421. doi:10.1109/ICNC47757.2020.9049740.

[66] Upstream, 2022. Upstream's 2022 global automotive cybersecurity report. URL: https://upstream.auto/2022report/.

[67] Wang, B., Sun, Y., Sun, M., Xu, X., 2021a. Game-Theoretic Actor–Critic-Based Intrusion Response Scheme (GTAC-IRS) for Wireless SDN-Based IoT Networks. IEEE Internet of Things Journal 8, 1830–1845. doi:10.1109/JIOT.2020.3015042.

[68] Wang, Y., Wang, Y., Qin, H., Ji, H., Zhang, Y., Wang, J., 2021b. A Systematic Risk Assessment Framework of Automotive Cybersecurity. Automotive Innovation 4, 253–261. doi:10.1007/s42154-021-00140-6.

[69] Wolf, M., Weimerskirch, A., Paar, C., 2004. Security in automotive bus systems, in: Proceedings of the workshop on Embedded Security in Cars (ESCAR)'04.

[70] Wright, S., 2021. Autonomous cars generate more than 300 tb of data per year. Tech Blog, Tuxera, Finland. URL: https://www.tuxera.com/blog/autonomous-cars-300-tb-of-data-per-year/.

[71] Xia, S., Qiu, M., Liu, M., Zhong, M., Zhao, H., 2019. AI Enhanced Automatic Response System for Resisting Network Threats, in: Qiu, M. (Ed.), Smart Computing and Communication, Springer International Publishing, Cham. pp. 221–230.

[72] Yarygina, T., Otterstad, C., 2018. A Game of Microservices: Automated Intrusion Response, in: Bonomi, S., Rivière, E. (Eds.), Distributed Applications and Interoperable Systems, Springer International Publishing, Cham. pp. 169–177.

[73] Zonouz, S.A., Khurana, H., Sanders, W.H., Yardley, T.M., 2014. RRE: A Game-Theoretic Intrusion Response and Recovery Engine. IEEE Transactions on Parallel and Distributed Systems 25, 395–406. doi:10.1109/TPDS.2013.211.

Biographical Sketches

Mohammad Hamad. He has been a research group leader with the Embedded Systems and Internet of Things Group at the Faculty of Computer Engineering, Technical University of Munich, Munich, Germany since 2020. He received his B.Eng. degree in Software Engineering and Information Systems from Aleppo University, Aleppo, Syria, in 2009. He also earned his Ph.D. (Dr.-Ing.) degree in Computer Engineering from the Institute for Data Technology and Communication Networks, Technical University of Braunschweig, Braunschweig, Germany, in 2020. His research interests lie in the area of autonomous vehicles and IoT security.

Andreas Finkenzeller. He received the B.Sc. and M.Sc. degrees in electrical engineering and computer science from Technical University Munich, Munich, Germany, in 2018 and 2021, respectively, where he is currently pursuing the Ph.D. degree with the Embedded Systems and Internet of Things Group. His research interests include embedded systems, secure communication, and IoT Security.

Michael Kühr. He received a B.Eng. degree in Electrical Engineering from the Baden-Wuerttemberg Cooperative State University in Stuttgart, Germany, in 2017 and a M.Sc. degree in Electrical Engineering and Information Technology from the Technical University of Munich, Munich, Germany, in 2022. His research interest focuses on the development and security of automated vehicles.

Andrew Roberts. He received the MCyberSecOps from University of New South Wales, Canberra, Australia in 2018 and the MSc degree in cybersecurity engineering from Tallinn University of Technology in 2020. He is currently pursuing the Ph.D. degree in information technology with the Tallinn University of Technology, Estonia. His current research is focussed on cybersecurity testing approaches to autonomous driving algorithms and methods to improve robustness of the design of autonomous systems to cyber threats.

Olaf Maenne. He got his PhD from the Technical University in Munich, studying wide-area Computer Networks and Network security through active and passive measurements and large-scale experiments. He has since then held faculty positions at Loughborough University in England and Tallinn University of Technology (TalTech) in Estonia, where he led the research at the Centre for Digital Forensics and Cybersecurity and established a Centre for Maritime Cybersecurity in Estonia. Since 2023, he has been with the University of Adelaide. His research interests have broadened over the years to include cyber defence technical exercises and critical infrastructure protection. He has been chairing numerous conferences, including ACM SIGCOMM in London in 2015 and the ACM Internet Measurement Conference (2017), and he is treasurer at ACM SIGCOMM 2024 in Sydney.

Vassilis Prevelakis. He received the B.Sc. degree (Hons.) in mathematics and computer science and the M.Sc. degree in computer science from the University of Kent, Canterbury, U.K., in 1984 and 1986, respectively, and the Ph.D. degree in computer science from the University of Geneva, Geneva, Switzerland, in 1996. He has worked in various areas of security in Systems and Networks both in his current academic capacity and as a freelance consultant. He is the Professor of Embedded Computer Security with the Technical University of Braunschweig, Braunschweig, Germany. His current research involves issues related to vehicular automation security, secure processors, security aspects of software engineering, and auto-configuration issues in secure VPNs.

Sebastian Steinhorst. He received the M.Sc. (Dipl.-Inf.) and Ph.D. (Dr. phil. nat.) degrees in computer science from Goethe University Frankfurt, Frankfurt, Germany, in 2005 and 2011, respectively. He is an Associate Professor with the Technical University of Munich, Munich, Germany, where he leads the Embedded Systems and Internet of Things Group, Department of Electrical and Computer Engineering. He was also a Co-Program PI in the Electrification Suite and Test Lab of the research center TUMCREATE in Singapore. The research of Prof. Steinhorst centers around design methodology and hardware/software architecture co-design of secure distributed embedded systems for use in IoT, automotive and smart energy applications.

Author Contributions

Mohammad Hamad: Conceptualization, Methodology, Writing – original draft, Funding acquisition, Supervision. Andreas Finkenzeller: Methodology, Writing – review & editing. Michael Kühr: Conceptualization, Methodology, Investigation, Software. Andrew Roberts: Validation, Writing – review & editing. Olaf Maenne: Validation, Writing – review & editing. Vassilis Prevelakis: Methodology, Validation, review & editing. Sebastian Steinhorst: Funding acquisition, Supervision, Writing – review & editing.