Toward a Multi-Layer Intrusion Response System for Connected Vehicles

Jan Lauinger, Mohammad Hamad, and Sebastian Steinhorst
Technical University of Munich
Munich, Germany
firstname.lastname@tum.de

***Abstract*—Cyber attacks are increasingly targeting connected vehicles. Due to this, Intrusion Response System (IRS) is becoming necessary to respond to unpreventable attacks. This paper proposes a multi-layer IRS prototype that incorporates distributed process management to support continuous control and monitoring of incident responses. Based on our analysis of IRS taxonomies, our prototype respects the latest IRS system requirements.**

***Index Terms*—Intrusion Response System, Incident Response, Automotive Intrusion Response System, Automotive Security, Connected and Autonomous Vehicles**

I. Introduction

The number of cyber-attacks on connected cars is increasing each year, despite the different security prevention and detection techniques that are implemented. Due to this, it becomes apparent that such techniques cannot adequately cover all vulnerabilities in Internet of Vehicles (IoV) ecosystems, making the use of IRS a necessity to ensure the safety and security of autonomous vehicles [1]. The ISO/SAE 21434 and UN Regulation No. 155 both emphasize the importance of having an incident response system as an integral part of automotive cyber-security. Such a system must be used to support the vehicle's manufacturer to respond to threats and vulnerabilities within, *a reasonable time frame*. Thus, autonomous incident response is essential for reducing response times, resolving cyber-attack effects, and preventing the spread of attacks.

As shown in Fig. 1, using information gathered by the Intrusion Detection System (IDS) and smart sensors, the response manager should respond to as many attacks as possible and support different response strategies. The collected information includes details about the cyberattack as well as contextual information about the vehicle. Until now, IRSs have received less attention and research efforts compared to IDSs. Consequently, attempts to implement IRSs in the automotive domain remain rare. In this paper, we analyze the latest IRS taxonomies to identify specific IRS requirements for the IoV domain (Sec. II). Next, we propose an IRS prototype design which respects all IRS requirements (Sec. III).

II. Background & Related Work

To overlook the complexity and requirements of the latest IRS, we analyze IRS taxonomies concerning applicability in the IoV. The comprehensive taxonomy of Stakhanova et al. [2] divides IRSs by *degree of automation* and *activity*. The *Degree of automation* considers types of responses and differentiates between manual, automated, and notification systems. *Activity* labels responses as active if they affect the actions of the attacker, otherwise responses count as passive. The taxonomy of Kanoun et al. [3] highlights temporal requirements of IRS where the feature of deactivation comes into play to stop the monitoring phase of ongoing responses. The work of Shameli-Sendi et al. [4] binds IRS taxonomy requirements in a consecutive processing model. Hamad et al. [1], [5] target specific IRS solutions in the IoV domain which introduce new requirements such as response efficiency and distribution.

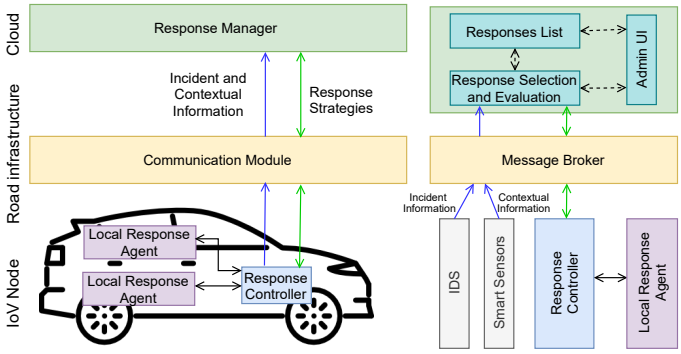

Fig. 1: System architecture of a multi-layer automotive IRS

By analyzing different IRS taxonomies, we identify the system requirements of Tab. I as important for an IRS in the IoV domain. Searching for our identified requirements in related works, we find that the work of Hoppe et al. [6] applies static response strategy mappings using a decision table. Based on attack severity data, information capacity, and the choice of the communication model, this work applies adaptive responses. The work of Nadeem et al. [7] applies active and passive adaptive responses. In contrast, our prototype allows monitoring and revocation of ongoing response actions.

Tab. I: IRS Taxonomy Requirements

System Requirement	Taxonomy Reference			
	[2]	[3]	[4]	[5]
Proactive/Reactive	●	●	●	●
Adjustability (static, dynamic)	●	●	●	●
Active/Passive	●	●	●	●
Mapping (static, dynamic, cost)	●	●	●	●
Efficient	○	○	◑	●
Distributed	○	○	○	●
Deactivation	○	●	●	●
Automation (cooperative, autonomous)	●	●	●	●

* ○ no support, ◑ partial support, ● full support

This work has received funding by the European Union's Horizon 2020 Research and Innovation Programme through the nIoVe project under grant agreement no. 833742.

III. IRS Architecture

A. Components

As shown in Fig. 1, IRS contains three main modules that are located in three different layers of the IoV ecosystem (i.e., In-vehicle, road infrastructure, and cloud). These modules are:

1) Response Manager: This module is located on the cloud. It is responsible for mapping suitable response actions to each security incident. This module contains three main components:

- Responses List: this includes all possible responses that can be implemented. these responses can include active responses (e.g., network reconfiguration and IP blocking) or passive responses (e.g., notification and logging).
- Response Selection and Evaluation: This part is responsible for selecting a response (or several) from the response list to deal with the reported cyber-security attack. A variety of input sources influence the selection of the response, including the incident and the vehicle's contextual information. The mapping method can be performed statistically by selecting the same response for the same attack type, or dynamically by choosing the most beneficial response with the least loss. In addition, this component monitors the execution of the selected response(s) and uses this information to adjust future selections or to stop certain responses.
- Admin UI: it is used by the system administrator to insert, update, and delete responses from the response list. Also, it can be used to visualize the status of responses implementation.

2) Communication Module: This module is used to exchange information between the Response Manager and the Response Controller. The communication module should meet different requirements, such as supporting one-to-many communication for the deployment of the same response among multiple vehicles at the same time if multiple vehicles are vulnerable to the same attack. Additionally, it should include mechanisms to ensure the integrity and confidentiality (as needed) of all messages.

3) Response Controller and Local Response Agents: Response controllers receive selected responses and distribute them to local response agents. Agents can be located on a specific Electronic Control Unit (ECU) or a domain gateway. Upon receiving a response from a controller, each agent implements it and reports its status back to that controller.

B. Software Prototype

The architecture of the software prototype is shown in Fig. 2. Within our prototype[1], the *Admin UI* was implemented using a web server that allows clients to send HTTP requests to the response manager Application Programming Interface (API) or to publish messages to the message broker. Events coming from the response manager are passed to browser clients via web-socket connections. The *Response List* is recognized as a database that stores responses.In addition, the database is used also to store return data and state information (e.g., response ID and revocation status). Celery Task Manager

[1]Source code: https://github.com/tum-esi/IRS

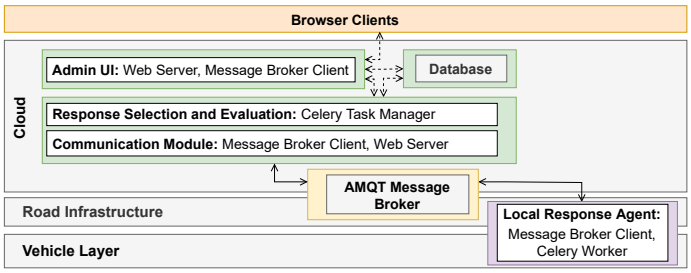

Fig. 2: Software Prototype of Automotive IRS

is used to implement the *Response Selection and Evaluation* and to handle incoming security incidents asynchronously and applies static and dynamic mappings of response strategies to distributed Celery[2] workers (*Response Agents*) via the message broker. Advanced Message Queuing Protocol (AMQP) Message Broker used to publish events to subscribed clients via different channels and to achieve scalable distribution of responses in the network. Here, it is possible to deploy message brokers as intermediaries on gateways or in cloud data centers. All services of the prototype support encapsulation for deployment in different container environments.

C. Evaluation

Parts of the prototype have been tested in the nIoVe[3] EU project where we set up a cloud manager in Greece (Thessaloniki) and a response controller in Germany (Munich). The evaluation results show end-to-end response times in the range of seconds considering up to 100 processes running concurrently.

IV. Conclusion

Based on the IRS taxonomy review, our work defines the logic of an IoV IRS protocol with the ability to continuously react and counteract security incidents manually and automatically. The software design focuses on non-blocking execution and provides scalable distributed process execution of response strategies using asynchronous workers. Our system supports a decision strategy which selects between active and passive responses.

References

[1] M. Hamad, "A multilayer secure framework for vehicular systems," Ph.D. dissertation, Technische Universität Carolo-Wilhelmina zu Braunschweig, 2020.

[2] N. Stakhanova, S. Basu, and J. Wong, "A taxonomy of intrusion response systems," *International journal of information and computer security*, vol. 1, no. 1-2, pp. 169–184, 2007.

[3] W. Kanoun, L. Samarji, N. Cuppens-Boulahia, S. Dubus, and F. Cuppens, "Towards a temporal response taxonomy," in *Data Privacy Management and Autonomous Spontaneous Security*. Springer, 2012, pp. 318–331.

[4] A. Shameli-Sendi, N. Ezzati-Jivan, M. Jabbarifar, and M. Dagenais, "Intrusion response systems: survey and taxonomy," *Int. J. Comput. Sci. Netw. Secur*, vol. 12, no. 1, pp. 1–14, 2012.

[5] M. Hamad, M. Tsantekidis, and V. Prevelakis, "Red-zone: Towards an intrusion response framework for intra-vehicle system." in *VEHITS*, 2019.

[6] T. Hoppe, S. Kiltz, and J. Dittmann, "Adaptive dynamic reaction to automotive it security incidents using multimedia car environment," in *2008 The Fourth International Conference on Information Assurance and Security*. IEEE, 2008, pp. 295–298.

[7] A. Nadeem and M. P. Howarth, "An intrusion detection & adaptive response mechanism for manets," *Ad Hoc Networks*, vol. 13, 2014.

[2]https://github.com/celery/celery

[3]https://www.niove.eu/