Graphical security modelling for Autonomous Vehicles: A novel approach to threat analysis and defence evaluation

Author

Nguyen, NH, Ge, M, Cho, JH, Moore, TJ, Yoon, S, Lim, H, Nelson, F, Bai, G, Kim, DD

Published

2025

Journal Title

Computers and Security

Version

Version of Record (VoR)

DOI

10.1016/j.cose.2024.104229

Rights statement

Downloaded from

Griffith Research Online

Computers & Security 150 (2025) 104229

Contents lists available at ScienceDirect

Computers & Security

journal homepage: www.elsevier.com/locate/cose

Full length article

Graphical security modelling for Autonomous Vehicles: A novel approach to threat analysis and defence evaluation

Nhung H. Nguyen [a,*], Mengmeng Ge [e], Jin-Hee Cho [b], Terrence J. Moore [c], Seunghyun Yoon [d], Hyuk Lim [d], Frederica Nelson [c], Guangdong Bai [a], Dan Dongseong Kim [a]

[a] School of Electrical Engineering and Computer Science, The University of Queensland, Brisbane, 4072, Queensland, Australia
[b] Department of Computer Science, Virginia Tech, Falls Church, VA, 22043, USA
[c] Network Science Division, US DEVCOM Army Research Laboratory, Adelphi, MD, 20783, USA
[d] AI Graduate School, Korea Institute of Energy Technology, Naju-si, 58217, South Korea
[e] Department of Computer Science and Software Engineering, University of Canterbury, Christchurch, 8041, New Zealand

ARTICLE INFO

Keywords:
Attack graphs
Autonomous Vehicle
Security analysis
Security modelling
Graphical security modelling
Attack countermeasure tree

ABSTRACT

Autonomous Vehicles (AVs) integrate numerous control units, network components, and protocols to operate effectively and interact with their surroundings, such as pedestrians and other vehicles. While these technologies enhance vehicle capabilities and enrich the driving experience, they also introduce new attack surfaces, making AVs vulnerable to cyber-attacks. Such cyber-attacks can lead to severe consequences, including traffic disruption and even threats to human life. Security modelling is crucial to safeguarding AVs as it enables the simulation and analysis of an AV's security before any potential attacks. However, the existing research on AV security modelling methods for analysing security risks and evaluating the effectiveness of security measures remains limited. In this work, we introduce a novel graphical security model and metrics to assess the security of AV systems. The proposed model utilizes initial network information to build attack graphs and attack trees at different layers of network depth. From this, various metrics are automatically calculated to analyse the security and safety of the AV network. The proposed model is designed to identify potential attack paths, analyse security and safety with precise metrics, and evaluate various defence strategies. We demonstrate the effectiveness of our framework by applying it to two AV networks and distinct AV attack scenarios, showcasing its capability to enhance the security of AVs.

1. Introduction

According to the Society of Automotive Engineers (SAE International, 2021), vehicle automation is divided into five levels. Levels 4 and 5 represent the highest level of automation, where Autonomous Vehicles (AVs) can operate and make decisions without human intervention. These levels require AVs to manage core vehicle functions, communicate with other vehicles and infrastructure through sensors, and integrate with external networks like satellites and cloud services.

A typical AV system architecture consists of several key components, including Electronic Control Units (ECUs), sensors, and communication buses. These components work together in a network to enable autonomous driving capabilities. Early automotive attacks primarily involved physical access (Koscher et al., 2010; Hoppe et al., 2011; Valasek and Miller, 2013), which target vehicles through interfaces such as the OBD-II port, USB, or CD player to control ECUs. These attacks ranged from manipulating the body control unit and door control unit to compromising critical ECUs like brake control, airbag system, and engine control. As AVs evolve, they introduce new vulnerabilities, which allow attackers to exploit remote access via wireless technologies (e.g., cellular, Wi-Fi, and Bluetooth) and sensors (such as RADAR, LiDAR, SONAR, and cameras). Research on remote attacks has uncovered vulnerabilities in information systems, autopilot features, and wireless communication (Checkoway et al., 2011; Miller and Valasek, 2015; Nie et al., 2017; Tencent Keen Security Lab, 2019), enabling attackers to escalate their attacks to more critical ECUs, raising significant concerns about AV security.

These vulnerabilities not only pose risks to individual components but also create new attack opportunities. Integrating new technologies into the infotainment system and advanced driver assistance systems creates network interfaces that allow attackers to interact with the AV's physical components. Such interactions could enable attackers to remotely inject malicious control commands, leading to loss of control and potentially causing accidents. Moreover, the AV system is a

* Corresponding author.
E-mail address: hongnhung.nguyen@uq.edu.au (N.H. Nguyen).

https://doi.org/10.1016/j.cose.2024.104229
Received 7 August 2024; Received in revised form 11 November 2024; Accepted 23 November 2024
Available online 30 November 2024

complex one, which is assembled from a variety of components sourced from original equipment manufacturers (OEMs) and supplemented with in-house developed software. This combination of components from various sources introduces an additional layer of complexity and expands the attack surface. Consequently, vulnerabilities in different parts of the vehicle could create new attack paths that are difficult to predict and mitigate.

Our research is motivated by the urgent need to address these issues. Due to the expensive and risky nature of security testing for AVs, we shift our focus towards security modelling and analysis as a proactive measure. However, there is a lack of research employing well-defined methods and metrics to model vehicle security, particularly at the system level, which considers the integration of diverse components. The current research landscape in AV security modelling primarily focuses on threat modelling, attack modelling, and qualitative security approaches. Yet, the systemic integration of AV components and quantitative measures remains under-researched. This gap prompts the following research questions: (1) How can we develop methodologies to model AV security and identify potential attack paths within AV systems, considering the complex integration of various components?; (2) Is it possible to quantitatively assess the security of an AV system, and if so, what metrics can effectively analyse and evaluate this security?; (3) What are the most effective defencive measures that can be implemented in AV systems, and how can these measures be determined?

A zero-day vulnerability is an unknown flaw in software or hardware that poses significant security risks as it can be exploited before a fix is available. In an AV network, the increasing complexity of software and the integration of electronic and mechanical systems heighten the risk posed by zero-day vulnerabilities. Such vulnerabilities in AVs can lead to severe security and safety breaches. In traditional computer networks, zero-day vulnerabilities mainly pose security risks. However, in AVs, these vulnerabilities can threaten both the security of the vehicle and the physical safety of its passengers, other road users, and pedestrians. Recognizing the critical issue of zero-day vulnerabilities in physical systems like AVs, we integrate our approach with "k-zero day safety" (Wang et al., 2014) analysis to identify the most critical components in AV systems and the number of zero-day vulnerabilities required to compromise the entire system. Additionally, we have developed a tool to automate the analysis of security vulnerabilities. Specifically, the AV-HARM tool processes input from the in-vehicle network architecture and attacker profiles to automatically generate visual graphs of potential attack paths, facilitating the analysis and quantification of the system's overall security.

The key contributions of this paper are summarized as follows:

- We propose the Autonomous Vehicle AV Hierarchical Attack Representation Model (AV-HARM) model to analyse the security of AV systems. This model identifies potential attack paths and associated risks within integrated AV systems. It enables the evaluation of various attack scenarios and the corresponding defences that can mitigate these threats.
- We introduce specific metrics to assess the security and safety of AV systems. These metrics serve as tools to quantify security at different levels, considering both known vulnerabilities and potential future threats. The metric also considers the effectiveness of the mitigation event for exploiting each vulnerability. It is effective in cases where the mitigation cannot eliminate the vulnerability.
- We build a web application to visualize the attack model and calculate security metrics automatically. The input data is the architecture of an AV system, which is stored in JSON format.
- We present a case study involving the system model, attack model, and defence model of an AV system, highlighting real-world vulnerabilities identified in these systems. We conduct a comparative analysis to evaluate various attack scenarios and defence methods. This case study aims to explain and validate our security models, demonstrating their application in a practical, real-world scenario.

The structure of this paper is as follows. Section 2 discusses related work on existing AV security models and the research gap. Section 3 demonstrates our model with the workflow, formal definitions, and metrics of the AV-HARM model. The model is evaluated using two vehicle networks presented in Section 4. Section 5 presents a comparative analysis of different vehicle systems, attack, and defence models. Section 6 outlines the limitations of our work and suggests future directions. Finally, Section 7 concludes the paper.

2. Related work

The field of autonomous vehicle security has been rapidly evolving, with many standards and studies addressing cybersecurity challenges. This section discusses existing work on cybersecurity standards and models for autonomous vehicles.

2.1. Security models for AV

EVITA (Ruddle et al., 2009) was a project funded by the European Commission aimed at enhancing AV security. This project utilized attack trees, a classification system, and a risk matrix to evaluate potential threats and security needs of vehicle assets. The attack trees outlined the goals, attack objectives, attack methods, and attack assets. These attack objectives helped assess severity, while the attack methods and assets were used to evaluate the likelihood of specific threats. The classification system was used to rate security severity using four vectors: safety, privacy, financial impact, and operational performance. The risk matrix integrated the probability of a successful attack, the severity of potential impacts, and the controllability of safety hazards to assign risk levels. The EVITA project is significant due to its comprehensive approach to cybersecurity risks in the automotive industry. However, the absence of a well-defined evaluation process may cause confusion for users (Luo et al., 2021). Additionally, the process relies heavily on expert judgment, lacking a clear method for determining values for each key element in the evaluation process.

In 2017, the European Telecommunications Standards Institute (ETSI) published the Technical Report TR 102 893 (ETSI Technical Committee ITS, 2017) titled "Intelligent Transport System (ITS); Security; Threat, Vulnerability and Risk Analysis (TVRA)". This report is a crucial part of the standards and guidelines for ITS, focusing on in-vehicle, vehicle-to-vehicle, and vehicle-to-roadside networks. The TVRA outlines security objectives, requirements, potential threats, risks, and countermeasures for ITS. These elements are structured around five key security attributes of CIAAA — confidentiality, integrity, availability, accountability, and authenticity. Risk assessment for each threat group in TVRA is based on two main attributes: impact and likelihood of attack. The latter considers potential attack factors such as time, expertise, knowledge, opportunity, and equipment. While the report provides a comprehensive list of potential threats, vulnerabilities, risks, and countermeasures, it is not exhaustive and lacks clarity in the methodology for identifying each value.

HEAVENS (Healing Vulnerabilities to Enhance Software Security and Safety) project (Lautenbach et al., 2021) presents a structured method for analysing cyber threats and evaluating risks within automotive systems. The central element of the HEAVENS security model is a risk matrix. This matrix determines the final Security Level (SL) based on the assessed Threat Level (TL) and Impact Level (IL). The IL includes several aspects: safety, financial, operational, privacy, and legislation, each with its detailed valuation criteria. The model also provides comprehensive definitions and explanations for assigning values to each TL and IL attribute. Moreover, it links specific security attributes to corresponding threats, using the STRIDE threat model (Kohnfelder and Garg, 1999) to categorize and assess the potential impact of various attacks. While the model is comprehensive and offers a high-level risk assessment for vehicles, it could benefit from further development in mitigation method evaluation and procedures for addressing specific vulnerabilities.

Monteuuis et al. (2018) proposed a comprehensive SARA framework for systematically analysing threats and assessing the risk of AV systems. This framework considers a broad range of factors influencing the network security of the AV system. These include human elements, vehicle control capacity, infrastructure sign recognition, and attack severity factors. They also expanded the STRIDE threat model to STRIDELC, introducing Linkability (the capacity to link anonymous data to identify its owner) and Confusion (authentic data structures with incorrect content, like damaged road signs). Like EVITA, the SARA framework also incorporates a risk matrix function that factors in controllability, attack severity, and attack likelihood.

Behfarnia and Eslami (2018) proposed a method for quantitatively assessing the risk of AVs. Instead of using attack graphs, they suggested the use of defence graphs. Unlike attack graphs, the nodes in defence graphs represent countermeasure methods. They devised a security model in which vulnerable components can be monitored through their security states. To calculate the likelihood and risk, they employed two methods: the Common Vulnerability Scoring System (CVSS) and EVITA. They conducted a case study on GPS spoofing to evaluate their model. This study allowed them to compare the system risk when implementing various anti-spoofing techniques on GPS, such as authentication, timing check, signal processing and amplitude/power monitoring.

Karray et al. (2018) introduced a comprehensive model for analysing the security of AV networks. This model includes multiple graph models: architecture graph, transformation graph, attack graph, and attack tree. They defined an architecture graph with four types of nodes: service, hardware, communication, and data. They proposed a Graph Transformations System (GTS), which was designed to capture all states during data and memory movements and changes. Using the GTS technique, the initial state data, and a set of rules, the authors generated a GTS for the entire network when a rule was applied. The model is comprehensive, but it has limitations. Specifically, while the GTS effectively illustrates data flows, its method of displaying the state is redundant and can be unwieldy. This could potentially affect scalability when applied to real-world vehicles with multiple systems.

Petho et al. (2021) introduced a method for analysing the security vulnerability of in-vehicle network topology. They modelled the in-vehicle network architecture using an un-directed weighted graph and determined the shortest path with Dijkstra's algorithm. The security vulnerability of in-vehicle networks in 15 different cars was evaluated. Their study found that the topology of the vehicle can reduce the effectiveness of the attack, increasing security and decreasing vulnerability values. They also compared their method with other research studies that do not use graphs but analyse the vehicle's topology using pentesting and statistics, concluding that their method yielded comparable results.

Khalid Khan et al. (2022) proposed a comprehensive system model to demonstrate the relationship between various security factors impacting a vehicle's security. The model impressively considers a broad range of variables, including in-vehicle network communication, physical access, human factors, trust within the industry, and public perception. They used a system dynamic model to show the relationships between elements, parameters, and factors, allowing users to analyse security in a complex context. The study also presents four archetypes to illustrate the model using qualitative analysis, demonstrating the impact of different scenarios such as hacker capabilities, human awareness, and regulatory laws. While the study is comprehensive, it remains a conceptual model that requires deeper analysis.

In summary, the studies mentioned above use various techniques to model and assess the security of AV systems, using both qualitative and quantitative methods. While the standard approach to AV risk assessment is generally broad, they use different terms and attributes to determine risk levels, typically relying on a risk matrix that considers the impact and likelihood of a successful attack. However, most of these studies do not incorporate real-world vulnerabilities in their analysis and lack full coverage of critical elements, such as the threat agent model, system model, attack model, defence model, and security metrics. This makes practical application challenging. Our model, AV-HARM, addresses these research gaps. It considers the system model, attack model, and defence model. Moreover, AV-HARM provides formal definitions, attacker goals, capabilities, and methods; a structured AV network model; a graphical method for attack and defence modelling; and multi-layered security metrics to assess AV security.

Our model integrates CVSS for vulnerability assessment but goes beyond individual vulnerability scoring and serves as a comprehensive security framework for AV networks. Unlike studies that rely solely on CVSS for system risk assessment, AV-HARM evaluates risk across the entire AV network. It considers a wide range of factors, including diverse threat scenarios, architectures, and defensive measures. For example, Behfarnia and Eslami (2018) and Jungebloud et al. (2024) use CVSS to assess AV vulnerabilities. However, Behfarnia and Eslami (2018) do not model the entire vehicle network or offer multi-layered risk metrics, limiting their evaluation to isolated vulnerabilities. Jungebloud et al. (2024) apply CVSS in an in-vehicle network case study but lack AV-specific security features such as safety impact, detection, mitigation events, and critical component identification through layered metrics.

In contrast, AV-HARM provides a comprehensive security model that includes essential elements such as safety, detection likelihood, and defence effectiveness, enabling a broader and more adaptable risk assessment tailored to AV systems.

In Table 1, we contrast our work with related studies using multiple criteria, including the Threat Agent Model, System Model Description, Attack Model Formalism, Defence Model Description, Metrics and Evaluation. The "Attacker factors" in the "Threat Agent Model" category include time, expertise, knowledge, opportunity, and equipment.

2.2. Applications of HARM on other networks

The initial Hierarchical Attack Representation Model (HARM) (Hong and Kim, 2012) was a two-layer security model designed to evaluate the cybersecurity of enterprise networks. This model integrates both attack trees and attack graphs to construct a comprehensive model for assessing a network's security risk. The attack graph represents all possible attack paths to the network, while the attack tree illustrates potential attack methods. Due to its flexibility, the HARM model can be applied in various domains, and numerous works have extended HARM to cater to different network characteristics.

In 2017, Ge et al. (2017) proposed a framework for analysing the security of Internet of Things (IoT) systems. This framework expanded the original HARM model into three layers, accommodating formal definitions specific to IoT systems and a set of security metrics. The application of this model was illustrated in detail for three different IoT scenarios: smart home, healthcare monitoring, and environmental monitoring systems.

Enoch et al. (2021) introduced a security model for maritime vessel networks, called MV-HARM. This model, an extension of the original HARM, includes definitions and adjustments tailored to vessel networks. The model also employs the STRIDE threat model to categorize the threats associated with each vulnerability and to compare different countermeasure methods for maritime vessel networks.

While AV-HARM shares its foundational structure with the earlier HARM adaptations, it distinctively tailors to the specific challenges of in-vehicle networks. Unlike its predecessors, AV-HARM identifies known vulnerabilities, emerging threats, and safety attributes that are crucial in the rapidly evolving automotive sector. Moreover, AV-HARM enhances the traditional attack tree model by incorporating an attack

Table 1
Comparison of different security analysis models.

Related work	Threat agent model	System model description	Attack model formalism	Defence model description	Metrics	Evaluation
Ruddle et al. (EVITA) (Ruddle et al., 2009)	Attacker factors	System assets (only mentioned)	Attack tree	Missing	Attack potential, threat severity, risk level, safety severity	Risk analysis of 11 use cases
ETSI et al. (TVRA) (ETSI Technical Committee ITS, 2017)	Attacker factors	List of functional assets	List of threats, vulnerabilities, consequences of threats	List of possible countermeasures	Risk metrics	Missing
Islam et al. (HEAVENS) (Lautenbach et al., 2021)	Attacker factors	List of assets (only mentioned)	Threat model (STRIDE)	Missing	Threat level, impact assessment, security risk	Evaluation based on 2 use cases
Monteuuis et al. (SARA) (Monteuuis et al., 2018)	Attacker profile and attacker capability	Missing	Threat models (STRIDELC), attack tree	Missing	Attack potential, attack goal severity, and risk	Evaluation based on 2 use cases
Behfarnia and Eslami (2018)	EVITA	Missing	Bayesian defence graphs	Missing	Impact and likelihood of threats	Case study with GPS component
Karray et al. (2018)	Missing	Architecture graph	Transformation graph, attack graph, and attack tree	Attack tree	Security risk	Missing
Petho et al. (2021)	Missing	IVN network topology	Undirected weighted graph	Missing	Shortest path and Security level	Analysed of 15 vehicles
Khalid Khan et al. (2022)	Human factors	Dynamic system model	Missing	Missing	Missing	System archetypes in 4 cases
AV-HARM (Our study)	Attacker goals, capabilities, and methods	AV network definition	Attack graphs and attack trees	Attack Countermeasure tree	Attack success probability, attack impact, risk on multiple levels, path-based metrics, and k-zero day safety	Analysed of 2 vehicle models and comparative analysis

countermeasure tree. This extra feature offers a way to evaluate the efficacy and impact of different defence strategies. This comprehensive and forward-thinking design makes AV-HARM especially effective in protecting vehicle networks from both existing and future threats.

3. Proposed security model

In this section, we introduce the AV-HARM security model and assessment. We will cover the model's workflow, attributes of AV systems, a formal definition of the security model, metrics, and methods to obtain the value for each security metric and element.

3.1. AV-HARM workflow description

As shown in Fig. 1, AV-HARM workflow contains four phases:

(1) Network Setup: The initial phase includes providing the in-vehicle network architecture, defining the threats to the network, and identifying vulnerabilities needed for the threat actors to compromise the system. This phase also outlines possible defence methods used in the in-vehicle system. For known vulnerabilities, it is possible to use vulnerability scanners and tools, such as Acunetix, OpenVAS, Nessus or Metasploit framework, to identify potential risks, as some AV components share similarities with computer networks. More details on the in-vehicle network architecture in AV-HARM are presented in Section 3.2.

(2) Security model: In this phase, the information about the in-vehicle network and vulnerabilities from the previous phase is used to generate the AV-HARM security model and visualize the attack graphs and attack trees. The security metrics at different system levels are also

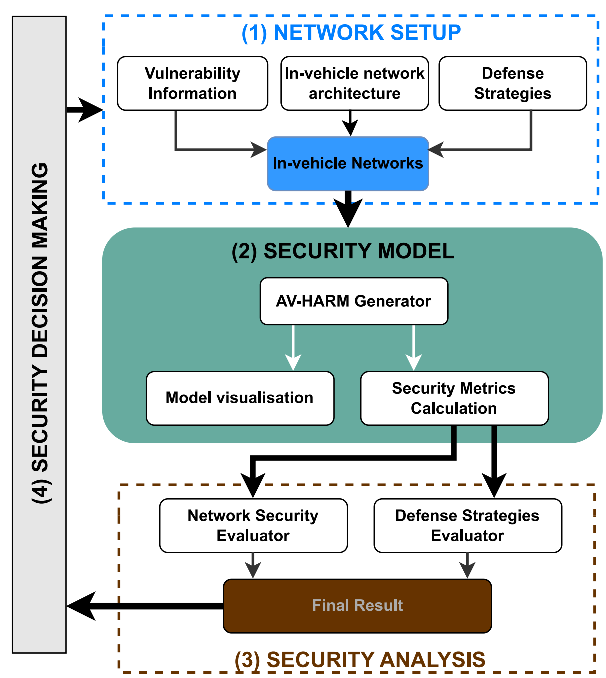


Fig. 1. The workflow of AV-HARM.

automatically calculated in this phase. More details on the AV-HARM model and security metrics are presented in Sections 3.3 and 3.4.

(3) Security analysis: In this phase, the security decision-makers can evaluate the metrics of the previous stage to conduct a system security risk analysis and determine the most effective and suitable defence methods within the system.

(4) Security Decision Making: In this phase, the security decision-makers can revise the defence components within the Network Setup, and/or adjust the information on vulnerabilities and/or the network architecture. The framework can be executed repeatedly until the goal for system security and safety is met. This methodology facilitates ongoing enhancement and evaluation of the efficacy of a range of security strategies, as well as the risk posed by different cyber threats, vulnerabilities and attack actions.

3.2. Attributes of the in-vehicle networks of an AV

In AV-HARM, an in-vehicle network of an AV is defined as follows:

An In-Vehicle Network IVN is characterized by three major attributes which are a finite set of Electronic Control Units ECU, a finite set of connecting buses that connect the ECUs BUS, and a finite set of vulnerabilities V. The details attributes of an $IVN = (ECU, BUS, VUL)$ are shown as follows:

Each electronic control unit $ecu \in ECU$ has a name ecu_{name}; a type $ecu_{type} \in \{diagclient, diagserver, gateway, normal\}$, where a $diagclient$ is an ECU that integrates with a diagnostic client which makes it possible to send diagnostic requests and control to other ECUs, an $diagserver$ is an ECU that integrates with a diagnostic server which primarily receives and responds to the requests and commands sent by the $diagclient$, $gateway$ ecu is the ecu present for the gateway function, and $normal$ is the normal ecu which only has it owns automotive function in a vehicle, respectively; a set of bus $ecu_{BUS} \subseteq BUS$ to which an ecu is connected; and a finite set of components ecu_C that comprises an ecu.

Each component $c \in ecu_C$ has a name c_{name}, an operating system c_{os}, a finite set of connected components $c_{connection}$ that can communicate with the component c, and a finite set of services c_S that runs inside the c_{os}.

Each service $s \in c_S$ has a name s_{name}; a finite set of privileges $s_{privilege}$ that run inside the c_{os}, if an attacker successfully compromises the service, they may gain these privileges; a finite set of connected services $s_{connection}$ that can communicate with service s; an attack surface $s_{attacksurface} \in true, false$ to show that this service has a connection to the external network; and a finite set of vulnerabilities s_V that is needed to compromise the service.

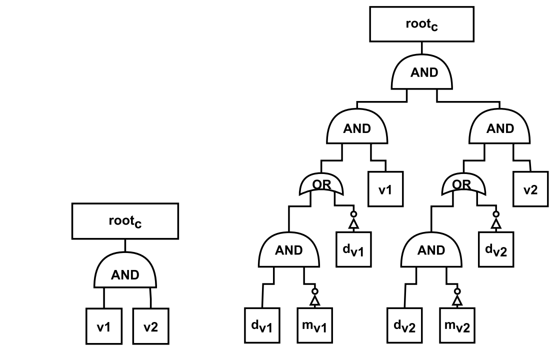
Each vulnerability $v \in VUL$ has a key v_{key}, a name v_{name}, a value of the probability of attack success v_{prob}; and a value of security impact v_{impact}, which is the impact of CIA if vulnerability v is exploited.

Each connecting bus $bus \in BUS$ has a key bus_{key}, a name bus_{name}, and a protocol $bus_{protocol} \in \{CAN, LIN, MOST, FlexRay, Ethernet\}$ that run on the bus.

3.3. AV-HARM security model definition

AV-HARM is a security model with three layers. The upper layer shows how attacks can move between the ECUs and their connections. The middle layer breaks down the ECUs into smaller parts, illustrating how an attacker might reach these components inside the ECUs. The lower layer details vulnerabilities and their logical relationships. This layer can include attack trees or attack countermeasure trees. Attack trees are used when no countermeasures are applied to vulnerabilities. However, when a defence method is implemented, Attack Countermeasure Trees (ACT) (Roy et al., 2010) are applied to evaluate security risks by considering detection events and assessing the effectiveness of mitigation strategies for each attack exploiting a specific vulnerability.

Definition 1 (*AV-HARM*)**.** The AV security model AV-HARM is defined as five tuple $AV - HARM = (U, M, L, C_{U\rightarrow M}, C_{B\rightarrow M})$. Here: U presents an upper layer which is an attack graph model; M presents a middle layer which is an attack graph model; L represents a set of attack countermeasure trees; $C_{U\rightarrow M}$ presents a mapping/connecting between the upper layer and the middle layer; and $C_{M\rightarrow L}$ presents a mapping/connecting between the middle layer and the lower layer.



(a) Attack tree without associated detection events or mitigation methods

(b) ACT with detection events and mitigation methods for each attack event using vulnerability

Fig. 2. Attack tree and ACT.

Definition 2 (*The Upper Layer*)**.** An upper layer of AV-HARM U is an attack graph that defined as two-tuple $U = (N_u, E_u)$, where $N_u \in ECU$ is a finite set of in-vehicle ECUs and $E_u \in BUS$ is a set of connections that connects the ECUs $E_u \subseteq N_u \times N_u$.

Definition 3 (*The Middle Layer*)**.** An middle layer of AV-HARM M is an attack graph defined as two-tuple $M = (N_m, E_m)$ where N_m is a finite set of components and E_m is a set of connections that connect the components $E_m \subseteq N_m \times N_m$.

Definition 4 (*The Lower Layer*)**.** A lower layer of AV-HARM, denoted by $L = \{at_1, at_2, at_3, \ldots, at_n\}$, comprises a set of attack trees where each tree, at_i, is associated with a specific node N_m^i. Each attack tree is defined by a five-tuple $L = \{V, G, c, gtp, root\}$, where: $V = c_{S_V}$ represents a set of vulnerabilities targeting the root privilege of node N_m^i; G is a set of logical gates including $\{AND, OR\}$ used in at_i; c is a mapping function that assigns gates to vulnerabilities or other gates; gtp denotes gate types, with $gtp \subseteq \{G \rightarrow \{AND, OR\}\}$; $root$ signifies the highest privilege level compromising node N_m^i.

Definition 5 (*The Attack Countermeasure Tree (ACT)*)**.** An ACT is linked to a specific node N_m^i in the middle layer and defined by a seven-tuple $L = \{V, D, MI, G, c, gtp, root\}$, where: V is a set of vulnerabilities an attacker might exploit; D is a set of detection events designed to identify attacks exploiting vulnerabilities $v_i \in V$; MI comprises mitigation methods to counter or prevent attacks exploiting vulnerabilities $v_i \in V$; G contains logical gates $\{AND, OR, NOT\}$ used in the ACT; c includes mappings such as $c(v_i) = AND(v_i, OR(NOT(d_i), AND(d_i, mi_i)))$ for a specific detection event $d_i \in D$ and a corresponding mitigation method $mi_i \in MI$; gtp is a gate type subset, where $gtp \subseteq \{G \rightarrow \{AND, OR\}\}$; and $root$ represents the highest privilege level to compromise node N_m^i.

Fig. 2 illustrates the attack tree and ACT used in our analysis. Case (a) shows an attack tree without detection or mitigation, while case (b) displays an ACT where each vulnerability is linked to a detection event and a corresponding mitigation method. Our model assumes a single mitigation method for each vulnerability to simplify the analysis.

3.4. Security metrics

In this section, we detail the security metrics used for AV-HARM. Similar to various AV security standards, our security metrics include security impact, probability of attack success, and security risk. However, we provide well-defined metrics with a clear explanation and framework to assign specific values to each metric. Our system includes four security metrics: attack success probability, attack impact, security risk, and k-zero score. The attack success probability, attack impact, and security risk are calculated across various hierarchical levels within

Table 2
List of AV-HARM security metrics with notations and explanations.

Parameters	Notations	Explanations
Attack success probability		
Vulnerability level	$prob_v$	Probability of a vulnerability is exploited after detection and defence
	$prob_v^{pre}$	Probability of a vulnerability is exploited before detection and defence
	$prob_v^{df}$	Effectiveness of defence against an attack exploiting the vulnerability
	$prob_v^{dt}$	Probability of detecting an attack that exploits the vulnerability
Component level	$prob_{comp}$	Probability of component being compromised
ECU level	$prob_{ecu}$	Probability of ECU is compromised
Attack path level	$prob_{ap}$	Probability of successful attack through attack path
System level	$prob_S$	Probability of entire system is compromised
Attack impact		
Vulnerability level	imp_v	Potential loss when vulnerability is compromised
Component level	imp_{comp}	Potential loss when component is compromised
	$impcia_{comp}$	Potential loss of CIA when component is compromised
	$imps_{comp}$	Potential loss of safety when component is compromised
ECU level	imp_{ecu}	Potential loss when ECU is compromised
Attack path level	imp_{ap}	Potential loss when an attack is successfully executed via the attack route
System level	imp_S	Potential loss when entire system is compromised
Security risk		
Component level	$risk_c$	The security risk when component is compromised
ECU level	$risk_{ecu}$	The security risk when ECU is compromised
Attack path level	$risk_{ap}$	The security risk of attack through attack path
System level	$risk_S$	The security risk of entire system
Path-based metrics		
Number of attack path	NAP	The total number of unique paths from an attacker to targets
Longest attack path length	$LAPL$	The length of the longest single attack path
k-zero safety		
System level	k	The number of zero-days that are needed to compromise the entire system

AV-HARM, from the lower layers to the upper levels. These metrics apply to different combinations of levels, such as the vulnerability level, component level, ECU level, path level, and the entire system level, depending on their relevance and applicability. Table 2 lists all the metrics, notations, and explanations of AV-HARM's security metrics.

3.4.1. Probability of successful attack

The attack success probability quantifies the likelihood of a threat successfully compromising a target. Targets can range from a vulnerability or a component to an ECU, an attack path, or an entire in-vehicle network.

In the lower level of AV-HARM, the probability of a successful attack at the root of an ACT is determined by the probability of exploiting a vulnerability at the leaf node. To estimate the probability of a vulnerability being exploited, it is possible to leverage the exploitability metrics provided by the Common Vulnerability Scoring System (CVSS) Base metric version 3.1 (FIRST.org Inc., 2019). These exploitability metrics are derived from four attributes: Attack Vector (AV), Attack Complexity (AC), Privileges Required (PR), and User Interaction (UI).

The base exploitability score from CVSS offers an initial estimation of a vulnerability's likelihood of being exploited. However, this estimation may be further refined by considering the presence of detection events and defence methods associated with the vulnerability. Using the Attack-Countermeasure Tree (ACT) in Fig. 2(b), the overall probability is calculated as shown in Eq. (1). Here, $prob_v^{pre}$ represents the pre-detection probability of a successful attack. Meanwhile, $prob_v^{dt}$ and $prob_v^{df}$ represent the probability of detecting an attack event exploiting the vulnerability and the effectiveness of the defence method applied to the attack event, respectively. The equation demonstrates that both detection and defence methods must be applied together to modify the initial probability of exploitation. Without either, the initial probability remains unchanged.

$$prob_v = prob_v^{pre} \times (1 - prob_v^{dt} \times prob_v^{df}) \tag{1}$$

Eqs. (2) and (3) illustrate the probability of a successful attack at the root of an ACT and the component, respectively.

$$prob_{root_i} = \begin{cases} 1 - \left(\prod_{v \in c(gtp_i)} (1 - prob_v) \right) & , \begin{smallmatrix} gtp_i \in G \\ c(gtp_i) = OR \end{smallmatrix} \\ \prod_{v \in c(gtp_i)} prob_v & , \begin{smallmatrix} gtp_i \in G \\ c(gtp_i) = AND \end{smallmatrix} \end{cases} \tag{2}$$

$$prob_{comp_i} = prob_{root_i} \tag{3}$$

Since the attack path and ECU in AV-HARM are composed of components, their attack success probabilities are defined based on the success probabilities of these components. Each node within an attack path or ECU is treated as an independent entity to simplify the analysis, enabling a modular assessment of each component's security. This approach allows decision-makers to evaluate the system's security at a granular level. Given that an attacker must compromise every node within the path to achieve their objective, the total success probability for the attack path is modelled as the product of the probabilities of compromising each node, reflecting a logical AND condition. The calculation of these metrics is shown in Eqs. (4) and (5).

$$prob_{ap_i} = \prod_{comp \in cap_i} prob_{comp} \tag{4}$$

$$prob_{ecu_i} = \prod_{comp \in ecu_i} prob_{comp} \tag{5}$$

The overall success probability of a network attack is calculated based on the attack paths in AV-HARM's middle layer, as depicted in Eq. (6). This equation shows the probability of a successful attack when the attacker can access the system via any of the attack paths. Each attack path from the attacker to the destination is considered independent.

$$prob_S = \max_{ap \in M_{AP}} (prob_{ap}) \tag{6}$$

3.4.2. Attack impact

The attack impact metric denotes the potential effect on the system if threat actors successfully compromise the target. Similar to the attack

Table 3
Safety criteria estimation.

Safety criteria at component ($imps_{comp_i}$)	Description
None (0)	No injuries
Minor (1)	Light and moderates injuries
Significant (2)	Severe and life-threatening injuries (survival probable)
Severe (3)	Life-threatening injuries (survival uncertain), fatal injuries
Catastrophic (4)	Life-threatening or fatal injuries for multiple vehicles

success probability metrics, the risk metrics are calculated at various levels. These include the node level, component level, ECU level, attack path level, and system level.

At the root level, the impact metric is calculated based on the effect of each vulnerability at the ACT's leaf node as shown in Eq. (7). The impact of a vulnerability can be obtained from the CVSS base score version 3.1, where it is determined based on its effect on confidentiality, integrity, availability, and scope change, which refers to the possibility of extending the scope to additional components.

$$imp_{root_i} = \max_{v \in c(gtp_i)} (imp_v) \tag{7}$$

The typical impact metric at the root level considers the security triad: Confidentiality, Integrity, and Availability. However, this CIA triad does not fully capture the complexities of physical systems like vehicles, where "Safety" is an equally important attribute. To assess the impact of a cyber attack on vehicles, it is necessary to incorporate the safety aspect into the evaluation process. Mapping the safety attribute directly to vulnerabilities is challenging because compromising one component may require multiple vulnerabilities combined using logic gates. Therefore, successfully exploiting one vulnerability may not affect the safety of the component(s). Our method estimates the safety impact only at the component level. To estimate the component's safety level, we recommend using the definitions according to ISO 26262 (International Organisation for Standardisation (ISO), 2018) and EVITA (Ruddle et al., 2009) as shown in Table 3. These standards rank safety based on the potential for severe injuries or vehicle damage in worst-case scenarios to ensure a comprehensive evaluation of risk. By considering the most severe possible outcomes, we capture the maximum impact that a compromised component could have on the vehicle and its occupants.

To calculate the entire impact of the component when it is compromised, we combine a safety impact factor with the other CIA factors. The weights w_{cia} and w_s reflect the relative importance of each term, as illustrated in Eq. (9). When combining the CIA attributes with the Safety attribute, we scale the impact on the CIA score, originally ranging from 0 to 6, to match the impact of the Safety score, which ranges from 0 to 4. This adjustment is achieved using Eq. (8).

$$impcia_{comp_i} = \frac{imp_{root_i} \times 4}{6} \tag{8}$$

$$imp_{comp_i} = w_{cia} \times impcia_{comp_i} + w_s \times imps_{comp_i} \tag{9}$$

As the component is part of an ECU, we calculate the impact of a cyber attack on the ECU as shown in Eq. (10). This is followed by the cyber attack impact on the attack path and at the system level, as depicted in Eqs. (11) and (12). Using the maximum operation in each equation is to make sure the impact of the cyber attack on each level ranges from 0 to 10.

$$imp_{ecu_i} = \max_{comp \in ecu_i} (imp_{comp}) \tag{10}$$

$$imp_{ap_i^M} = \max_{comp \in ap_i^M} (imp_{comp}) \tag{11}$$

$$imp_S = \max_{ap_i \in M} (imp_{ap^m}) \tag{12}$$

3.4.3. Security risk

The security risk metric quantifies the severity of the potential impact of a threat and the probability of its occurrence. We assess the security risk of an AV network across four hierarchical levels: the component level, the ECU level, the attack path level, and the entire system level. At the component level, risk is quantified by the probability of attack success and the subsequent impact on the component, as delineated in Eq. (13).

$$risk_{comp_i} = prob_{comp_i} \times imp_{comp_i} \tag{13}$$

At the ECU level, the security risk is derived from the cumulative risk of the individual components comprising the ECU, represented in Eq. (14). Similarly, at the attack path level, it is the sum of the security risks of components along the path, as shown in Eq. (15). Our approach considers cumulative risk: the more components an attacker must exploit, the greater the potential risk to the system. This is because each component, whether deemed critical or not, contributes to the overall security of the system. Compromising multiple components accumulates vulnerabilities and potential impacts, resulting in a compounded risk along the attack path.

$$risk_{ecu_i} = \sum_{comp \in ecu_i} risk_{comp} \tag{14}$$

$$risk_{ap^M} = \sum_{comp \in ap_i^M} risk_{comp} \tag{15}$$

The overall security risk of the AV network is characterized by combining the risk across all attack paths, from the threat to the attack goals. This comprehensive network risk is shown in Eq. (16):

$$risk_S = \sum_{ap^M \in \mathrm{M}} risk_{ap^M} \tag{16}$$

3.4.4. Path-based metrics

In addition to security metrics at various levels, path-based metrics such as the total number of distinct attack paths and the lengths of these paths are crucial for understanding the breadth and depth of potential attack routes through the system.

The Number of Attack Paths (NAP) quantifies the total number of unique sequences that an attacker can use to successfully reach a target. The metric is defined as in Eq. (17):

$$NAP = |AP^M| \tag{17}$$

where AP^M denotes the set of all possible attack paths within the middle layer of the model.

The Longest Attack Path Length ($LAPL$) is a metric that determines the maximum depth of attack required to reach a target. The longest attack path length is computed as in Eq. (18):

$$LAPL = \max_{ap \in AP^M} |ap| \tag{18}$$

3.4.5. k-zero score

While vehicle networks share similarities with computer networks, identifying and scanning vulnerabilities AV systems can be challenging for several reasons: (1) Vehicles consist of various components from different original equipment manufacturers (OEMs), making it difficult to assess and scan for all potential security vulnerabilities, (2) Many AV systems use proprietary software and protocols, restricting access to necessary information for vulnerability scanning, (3) Existing security solutions cannot detect unknown vulnerabilities.

Defining vulnerabilities is crucial for assessing AV networks. However, the risk posed by unknown vulnerabilities remains unpredictable. To measure this risk, we implement the concept of k-zero day, as

Result: k-zero day safety
Input: Network model and (zero-day vulnerability in nodes), Target components
Output: k-zero day safety
Procedure calculateK(*networkModel, targetComponents*)**:**
  Initialize an empty array to store the number of nodes on each attack path: numberofnodearr = [];
  Generate attack paths using the AV-HARM: drawAVHARM();
  attackPaths = findAttackPath(Attacker, targetComponents);
  if *attackPaths.count > 0* **then**
    foreach *attackPath in attackPaths* **do**
      numberofnode = getNumberOfNodeOnPath();
      numberofnodearr.push(numberofnode);
    end
  end
  Determine the minimum number of nodes across all attack paths: k = min(numberofnodearr);
return *k*

Algorithm 1: Calculate k-zero day safety for AV-HARM

proposed by Wang et al. (2014). This research aims to quantify the security risk of network systems against the uncertainty of unknown zero-day attacks by determining how many unknown vulnerabilities (k-zero day number) would need to be exploited to compromise network assets. A larger k value indicates a more secure network. In AV-HARM, we use this concept to determine how many k-zero days are required to compromise attack goals in AV networks. This is particularly useful when no known vulnerabilities are present, but the system's architecture still needs to be evaluated for potential zero-day vulnerabilities that would be needed to compromise the system. It also helps in evaluating the security of the system design under the assumption that each component node has at least one potentially exploitable vulnerability. The k value is defined by generating AV-HARM, identifying all attack paths from the attacker to the attack goals, and finding the shortest path that leads to a successful compromise attack goal. This represents the minimum number of vulnerabilities that need to be exploited, also known as k. The algorithm for calculating the k-zero day safety number is shown in Algorithm 1.

We built a web application for visualizing and automatically calculating security metrics. It is accessible via the link https://ziz0301.github.io/AVHARM/index.html. This web application uses Javascript and the GoJS library for drawing attack graphs and trees. The application accepts JSON file inputs that follow the in-vehicle network format in Section 3.2. From the network input, the application automatically generates attack graphs and trees, along with metric calculations.

4. Application of the AV-HARM to assess the AV networks

In this section, we further explore the AV-HARM framework introduced previously by applying it to an example AV network. We implement AV-HARM on two vehicle models: the BMW i3 and the Tesla Model S. We will discuss the system model, attack model, and defence model. Additionally, we will illustrate how to use the proposed security metrics to evaluate and quantify the security of the AV system in each case.

4.1. Case study of a BMW i3 vehicle

4.1.1. System model

In-vehicle network architecture: The in-vehicle network of a BMW i3 is referenced based on a BMW car security report (Cai et al., 2019; BMW, 2014; Jungebloud et al., 2024), as shown in Fig. 3. This partial in-vehicle network is a heterogeneous system with various chips and microprocessors running diverse operating systems and services. The car's connectivity capabilities are supported by two Electronic Control Units (ECUs): the Head Unit (HU) and the Telematic Communication Box (TCB), both connected to the Gateway (ZGM) using a KCAN bus. These units enable the car to connect to external networks and are the primary attack vectors for potential remote or physical attacks on the car's internal network. The HU and TCB are connected to the Gateway (ZGM) using the KCAN bus.

The HU consists of two main components, both running the QNX operating system: HU-Intel and HU-Jacinto. The HU-Intel is responsible for the car's infotainment system, including information display, navigation, diagnostic functions, and network connections. For security reasons, the HU-Intel cannot directly send arbitrary CAN messages to the ZGM. Instead, these messages must be sent to the ZGM through the HU-Jacinto.

Similar to the HU, the TCB also has two main components. The TCB-Qualcomm is responsible for all remote services to the manufacturer's server, including emergency calls, tele-diagnostic, remote services, and Over-the-Air (OTA) updates. The second component, TCB-FreeScale, manages sending and receiving CAN messages and is directly connected to the ZGM.

The ZGM connects to various CAN buses and a FlexRay bus. The KCAN is solely for entertainment and basic functions, including the ECU HU, TCB, Frontal Light Electronics Right (FLER) for controlling the vehicle's left and right lights, and Body Domain Controller (BDC) for managing other body functions like doors, windows, and wipers. The PTCAN is for critical functions, such as the Electrical Digital Motor Electronics (EDME) for throttle control, and the Electronic Power Steering (EPS) for managing the vehicle's steering.

Only ECUs on the same CAN bus can communicate with each other. However, the ZGM can receive diagnostic messages via the Unified Diagnostic Services (UDS) protocol from the diagnostic client (DC) in HU-Intel and TCB-Qualcomm. This means they can send diagnostic messages to any ECU with a diagnostic server (DS).

Threat and vulnerabilities information: We utilized known vulnerabilities from a real vehicle's security report (Cai et al., 2019) to illustrate the information in this example. The details of each vulnerability are shown in Table 4. This includes notation, the CVE-ID, the associated component, a description of the vulnerability, and the CVSS score. When a vulnerability lacks a designated CVE-ID or a CVSS 3.1 score, we provide the corresponding CVSS vector used to compute the score.

$$prob_{v_i^{pre}} = \frac{prob_{v_i^{CVSS}}}{3.9} \tag{19}$$

The CVSS 3.1 framework measures exploitability on a scale from 0 to 3.9. To align these metrics with our probabilistic calculations, which also incorporate logical AND/OR operations to find out the probability of an attack's success, a scaling adjustment to a 0 to 1 range is necessary. We use Eq. (19) to scale it accordingly. A value of 0 indicates that the vulnerability is difficult to exploit, whereas a value of 1 implies a high certainty of exploitability.

4.1.2. Attack model

In this section, we describe the attack model, each focusing on specific attacker goals, attack vectors, attack surfaces, attacker capabilities and methods.

Attacker Goal 1: Compromise the root privilege information system, which is the HU-Intel and TCT-Qualcomm infotainment system. There are two attack vectors to bypass the infotainment system. The first attack vector is via physical access using USB. The attacker inserts a USB containing a malicious update file for the navigation service. They then use v_{B3} to obtain root privileges on the Head Unit-Intel (HU-Intel) (Miller, 2019; Cai et al., 2019).

The second attack vector is via remote access by setting up a rogue base station to access the cellular network of the infotainment unit. From there, the attacker gains root privilege of TCB-Qualcomm using

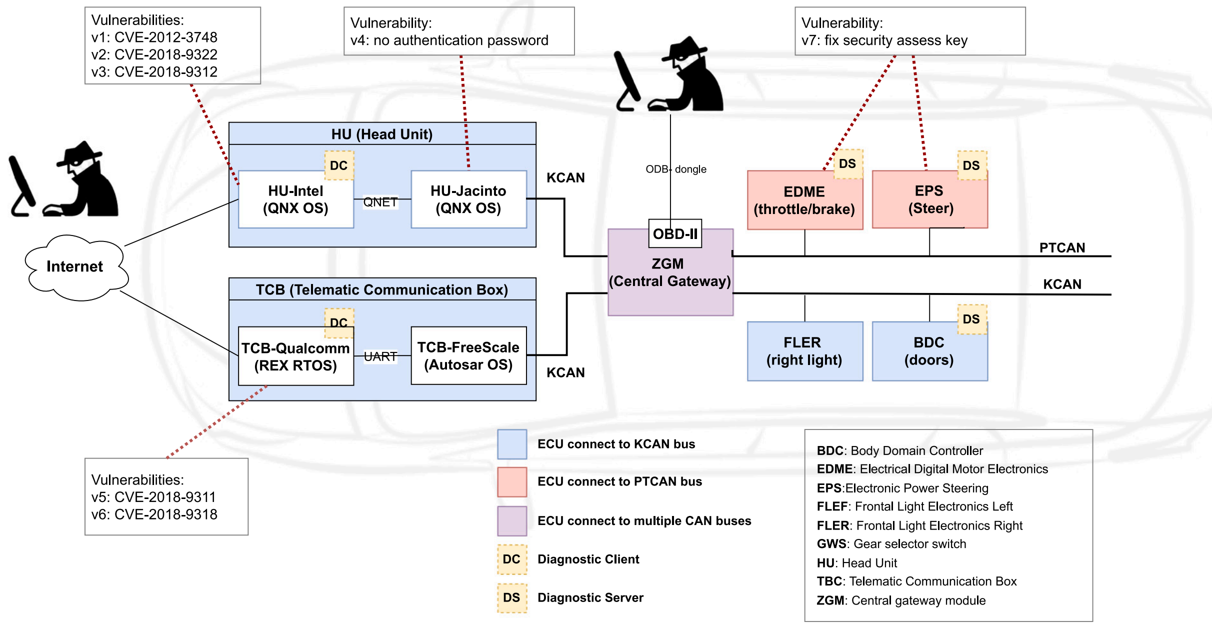


Fig. 3. In-vehicle network architecture of a BMW i3 vehicle.

Table 4
Vulnerabilities information within BMW i3 vehicle.

Notations	CVE-ID	Components	Description	CVSS 3.1
v_{B1}	CVE-2012-3748	HU-Intel	The browser runs an old version of AppleWebKit/535.17. It exploits memory corruption (CWE362) in the Webkit engine (libwebkit-hbas-NBT.so) to gain a shell with browser privilege	Base score: 7.7 CIA impact: 3.53, likelihood: 0.46 AV:N/AC:H/PR:L/UI:N/S:C/C:L/I:H/A:L
v_{B2}	CVE-2018-9322	HU-Intel	TOCTOU weakness (CWE367) in the diagnostic service (/opt/sys/bin/NbtDiagHUHighApp) leads to bash commands execution with HU-Intel root privilege	Base score: 7.7 CIA impact: 3.53, likelihood: 0.46 AV:N/AC:H/PR:L/UI:N/S:C/C:L/I:H/A:L
v_{B3}	CVE-2018-9312	HU-Intel	Stack overflow weakness (CWE121) in the navigation map update service (filename decompression function) (/opt/nav/asn/bin/appnavc) leads to code execution with HU-Intel root privilege	Base score: 7.8 CIA impact: 3.93, likelihood: 0.46 AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H
v_{B4}	No CVE-ID	HU-Jacinto	Weak authentication on QNX OS allows an attacker to log into HU-Jacinto without a password	Base score: 8.2 CIA impact: 3.13, likelihood: 0.72 AV:A/AC:L/PR:N/UI:N/S:C/C:N/I:H/A:L
v_{B5}	CVE-2018-9311	TCB-Qualcomm	Vulnerability in the remote service (TCB_Of_RemoteProcedureCallManager_Task) allows the attacker to send malicious SMS using a rogue base station	Base score: 9.8 CIA impact: 3.93, likelihood: 1 AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H
v_{B6}	CVE-2018-9318	TCB-Qualcomm	OTA update vulnerability. The stack overflow in the OTA_XML_VerifySignature function is exploited to gain remote execution with TCB-Qualcomm root privilege.	Base score: 9.8 CIA impact: 3.93, likelihood: 1 AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H
v_{B7}	No CVE-ID	ECUs with diagnostic server	The Security Access service of the UDS protocol uses weak algorithms. This enables an attacker to incorporate diagnostic functions and manipulate updates on the ECU.	Base score: 9.0 CIA impact: 4, likelihood: 0.59 AV:A/AC:L/PR:L/UI:N/S:C/C:H/I:H/A:H

v_{B5} and v_{B6}. The attacker then exploits a vulnerability in the HU-Intel's browser using v_{B1} to obtain a shell with browser privileges and leverages a Time-of-Check to Time-of-Use (TOCTOU) vulnerability in the diagnostic service for root access (Miller and Valasek, 2015; Cai et al., 2019; Nie et al., 2017)

Attacker Goal 2: Gain control of the ECU on the same CAN bus as the HU and TCB. Since the HU and TCB share the same KCAN bus with the FLER and BDC, the attacker can extend their attack from HU-Intel to HU-Jacinto. From there, they can send arbitrary CAN messages to the BDC and FLER. This attack vector presupposes that the attacker has deciphered the structure CAN message for controlling the BDC and FLER of this vehicle.

Attacker Goal 3: Control the ECU on a different CAN bus from the HU-Intel and TCB-Qualcomm. The HU and TCB are connected to the external network, and for security reasons, they are designed to only connect to non-critical components. Critical components are connected to different buses. Under normal circumstances, it is unrealistic for an attacker to send regular CAN messages to control critical ECUs, such as brakes or steering. However, the possibility of over-the-air updates and remote diagnostics gives the HU and TCB the ability to send UDS messages from the HU-Intel and TCB-Qualcomm to critical components on the PTCAN bus. In this attack scenario, the attacker can send UDS messages to manipulate the EDME and EPS, which are on the PTCAN bus. This attack vector assumes that the attacker has deciphered the structure of UDS messages to diagnose the brake and steering functions of the vehicle.

4.1.3. Applying AV-HARM to calculate security metrics of BMWi3

We use the AV-HARM application to generate the model and visualize the attack graphs of the upper layer and the middle layer and the attack trees at the lower layer as shown in Fig. 4. By modelling the vehicle architecture with the vulnerabilities information, we can calculate there are 8 possible attack paths, including:

- $ap_{B1} = A \rightarrow HU - Intel \rightarrow HU - Jacinto \rightarrow BDC$,
- $ap_{B2} = A \rightarrow HU - Intel \rightarrow BDC$,
- $ap_{B3} = A \rightarrow TCB - Qual \rightarrow BDC$,
- $ap_{B4} = A \rightarrow HU - Intel \rightarrow HU - Jacinto \rightarrow FLER$,
- $ap_{B5} = A \rightarrow HU - Intel \rightarrow EDME$,
- $ap_{B6} = A \rightarrow TCB - Qual \rightarrow EDME$,
- $ap_{B7} = A \rightarrow HU - Intel \rightarrow EPS$,
- $ap_{B8} = A \rightarrow TCB - Qual \rightarrow EPS$.

Based on the metric in Section 3.4, we calculate the probability of attack success, the attack impact and attack risk on vulnerability level, component level, ecu level, attack path level and system level. We will detail the calculation of the probability of attack success and attack impact on each component.

$$\begin{aligned}
prob_{HU-Intel} &= 1 - [(1 - prob_{v_{B3}}) \times (1 - (prob_{v_{B1}} \\
&\quad \times prob_{v_{B2}}))] \\
&= 1 - [(1 - 0.46) \times (1 - (0.46 \times 0,46))] \\
&= 0.57 \\
prob_{HU-Jacinto} &= prob_{v_{B4}} = 0.72 \\
prob_{TCB-Qual} &= prob_{v_{B5}} \times prob_{v_{B6}} = 1 \times 1 = 1 \\
prob_{BDC} &= prob_{FLER} = prob_{v_{B7}} = 0.59 \\
prob_{EDME} &= prob_{EPS} = prob_{v_{B7}} = 0.59
\end{aligned}$$

For assessing the impact of an attack, we assign equal weight to security and safety in our calculations, setting both w_{cia} and w_s to 1. Following the safety criteria estimation presented in Table 3, we assign a safety impact value of 1 to HU-Intel, HU-Jacinto, TCB-Qual, and FLER. This is because compromising these components may lead to minor injuries, such as altering vehicle lighting or displaying incorrect notifications or data on the HU or TCB unit. The BDC is assigned a value of 2 since if the door is open while the vehicle is in motion, it could cause severe injuries. The safety impact of EDME and EPS is given a value of 3, as issues with steering and throttle could lead to life-threatening injuries to the vehicle occupants or pedestrians.

$$\begin{aligned}
imp_{HU_Intel} &= w_{cia} \times impcia_{HU_Intel} + w_s \times imps_{HU_Intel} \\
&= 1 \times 3.93 + 1 \times 1 = 4.93 \\
imp_{HU-Jacinto} &= w_{cia} \times impcia_{HU-Jacinto} \\
&\quad + w_s \times imps_{HU-Jacinto} \\
&= 1 \times 3.13 + 1 \times 1 = 4.13 \\
imp_{TCB-Qual} &= w_{cia} \times impcia_{TCB-Qual} \\
&\quad + w_s \times imps_{TCB-Qual} \\
&= 1 \times 3.93 + 1 \times 1 = 4.93 \\
imp_{BDC} &= w_{cia} \times impcia_{BDC} + w_s \times imps_{BDC} \\
&= 1 \times 4 + 1 \times 2 = 6 \\
imp_{FLER} &= w_{cia} \times impcia_{FLER} + w_s \times imps_{FLER} \\
&= 1 \times 3.93 + 1 \times 1 = 4.93 \\
imp_{EDME} &= w_{cia} \times impcia_{EDME} + w_s \times imps_{EDME} \\
&= 1 \times 4 + 1 \times 3 = 7
\end{aligned}$$

$$\begin{aligned}
imp_{EPS} &= w_{cia} \times impcia_{EPS} + w_s \times imps_{EPS} \\
&= 1 \times 4 + 1 \times 3 = 7
\end{aligned}$$

Next, we calculate the risk of each component using Eq. (13), which considers both the probability of a successful attack and its impact.

$$\begin{aligned}
risk_{HU-Intel} &= prob_{HU-Intel} \times imp_{HU-Intel} \\
&= 0.57 \times 4.93 = 2.81 \\
risk_{HU-Jacinto} &= prob_{HU-Jacinto} \times imp_{HU-Jacinto} \\
&= 0.72 \times 4.13 = 2.97 \\
risk_{TCB-Qual} &= prob_{TCB-Qual} \times imp_{TCB-Qual} \\
&= 1 \times 4.93 = 4.93 \\
risk_{BDC} &= prob_{BDC} \times imp_{BDC} \\
&= 0.59 \times 6 = 3.54 \\
risk_{FLER} &= prob_{FLER} \times imp_{FLER} \\
&= 0.59 \times 4.93 = 2.91 \\
risk_{EDME} &= prob_{EDME} \times imp_{EDME} \\
&= 0.59 \times 7 = 4.13 \\
risk_{EPS} &= prob_{EPS} \times imp_{EPS} \\
&= 0.59 \times 7 = 4.13
\end{aligned}$$

The risk for each path and the overall system risk are calculated using Eqs. (15) and (16)

$$\begin{aligned}
risk_{ap_{B1}} &= risk_{HU-Intel} + risk_{HU_Jacinto} + risk_{BDC} \\
&= 2.81 + 2.97 + 3.54 = 9.32 \\
risk_{ap_{B2}} &= risk_{HU-Intel} + risk_{BDC} \\
&= 2.81 + 3.54 = 6.35 \\
risk_{ap_{B3}} &= risk_{TCB_Qual} + risk_{BDC} \\
&= 4.93 + 3.54 = 8.47 \\
risk_{ap_{B4}} &= risk_{HU-Intel} + risk_{HU_Jacinto} + risk_{FLER} \\
&= 2.81 + 2.97 + 2.91 = 8.69 \\
risk_{ap_{B5}} &= risk_{HU-Intel} + risk_{EDME} \\
&= 2.81 + 4.13 = 6.94 \\
risk_{ap_{B6}} &= risk_{TCB_Qual} + risk_{EDME} \\
&= 4.93 + 4.13 = 9.06 \\
risk_{ap_{B7}} &= risk_{HU-Intel} + risk_{EPS} \\
&= 2.81 + 4.13 = 6.94 \\
risk_{ap_{B8}} &= risk_{TCB_Qual} + risk_{EPS} \\
&= 4.93 + 4.13 = 9.06
\end{aligned}$$

$$\begin{aligned}
risk_S &= risk_{ap_{B1}} + risk_{ap_{B2}} + risk_{ap_{B3}} + risk_{ap_{B4}} \\
&\quad + risk_{ap_{B5}} + risk_{ap_{B6}} + risk_{ap_{B7}} + risk_{ap_{B8}} \\
&= 9.32 + 6.35 + 8.47 + 8.69 + 6.94 + 9.06 \\
&\quad + 6.94 + 9.06 \\
&= 64.83
\end{aligned}$$

The system risk for this network stands at 64.83, representing the risk level of the entire system. This value takes into account all possible attack paths an attacker could exploit to infiltrate the system. By assessing the risk of each potential attack path, security decision-makers can

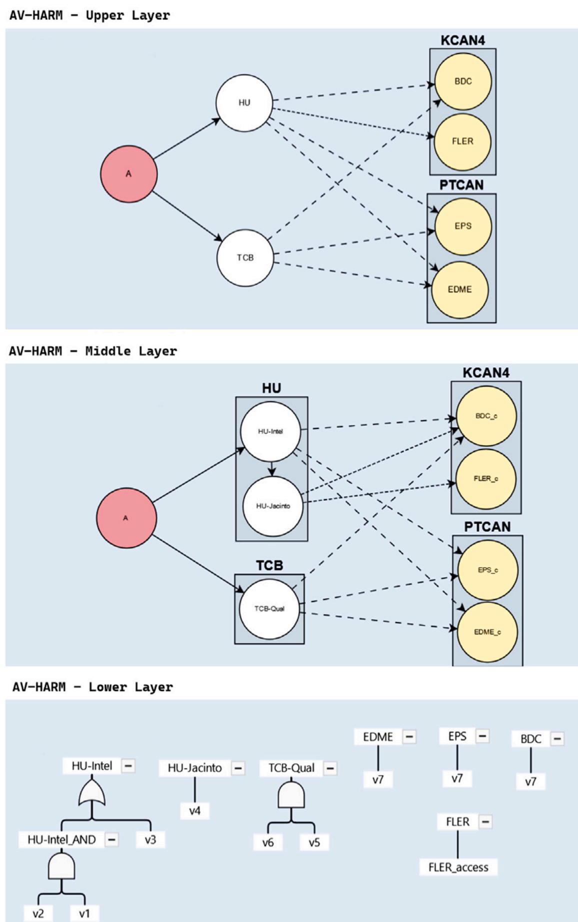


Fig. 4. Attack paths in AV-HARM for the BMW i3 vehicle.

identify the most threatening ones and adopt appropriate mitigation methods to diminish the risk or eliminate the attack path.

4.2. Case study of Tesla Model S vehicle

4.2.1. System model of a Tesla Model S vehicle

In-vehicle network architecture: The in-vehicle network of a Tesla Model S is referenced based on a security report by Keen Security Lab (Nie et al., 2017, 2018; Tencent Keen Security Lab, 2016), as depicted in Fig. 5. Similar to the BMW system, this partial in-vehicle network also consists of heterogeneous components running various operating systems and services.

In this vehicle network, the infotainment system ECU can connect to the external network via an embedded Wi-Fi SSID and be able to connect to a Wi-Fi hotspot provided by Tesla. The infotainment system includes three components: the Instrument Cluster (IC), the Central Information Display (CID) and the Parrot Model. Among those, the CID is running under the Linux kernel, which has services/software such as Browser, AppArmor and IPtables. These three components are connected via Automotive Ethernet.

To connect with the CAN bus domain, the infotainment system interfaces with a gateway. This gateway operates on firmware running on the Freescale MPC5668G microcontroller. In the BMW vehicle described earlier, the HU includes the HU-Jacinto component, which is responsible for sending CAN messages to the gateway. However, in Tesla vehicles, the infotainment system does not send CAN messages directly to the gateway; instead, it sends Ethernet messages. The gateway in Tesla vehicles is programmed to handle these Ethernet messages, converting them into CAN messages for transmission to other ECUs. Additionally, the gateway has the capability to send diagnostic messages with various available functions.

The CAN bus network system comprises two buses: the BDY bus and the PT bus. The BDY bus primarily controls the car's body functions, including the ECU that manages the trunk and door. Critical and safety ECUs, such as the Electronic Stability Program (ESP) and Anti-Lock Brake System (ABS), are connected to the PT bus.

Threat and vulnerabilities information: Table 5 shows the 5 vulnerabilities of the vehicle. Similar to the example of a BMW vehicle, the table shows the CVE-ID, description and the attribute that contribute to the CVSS scores.

4.2.2. Attack model of a Tesla Model S vehicle

In this section, we define the attack model, each focusing on specific attacker goals, attack vectors, attack surfaces, attacker capabilities and methods.

Attacker Goal 1: Compromise the infotainment system. The attacker sets up a fake Wi-Fi hotspot for the CID components to connect to. From this point, the attacker uses vulnerabilities v_{T1} and v_{T2} to gain shell access with root privileges to the CID. The attacker then exploits vulnerability v_3, a weak authentication vulnerability, to gain root access to the IC and Parrot components.

Attacker Goal 2: Compromise the gateway ECU. The gateway has a service used for sending diagnostic services and a backdoor to enable its root shell. However, the security of the backdoor is weak, with a fixed static password that can be discovered through reverse engineering, also known as v_{T3}.

Attacker Goal 3: Control the ECU on two different CAN buses: BDY and PT CAN, from the gateway and infotainment system. From the gateway, the attacker can send CAN messages to control the body functions of the vehicle, such as opening doors and turning the lights on and off. However, control over more critical functions is not possible, even when the attacker has access to the gateway. This is because some ECUs have implemented safety mechanisms to disable unsafe messages and stop responding to those messages to prevent dangerous operations. However, the attacker can use the diagnostic service from the IC to send UDS messages, causing the ECUs to malfunction.

4.2.3. Apply AV-HARM and calculate security metrics

We use the AV-HARM application to generate the model and visualize the attack graphs of the upper layer and the middle layer and the attack trees at the lower layer, as shown in Fig. 6. By modelling the vehicle architecture with the vulnerabilities information, we can calculate there are 6 possible attack paths, including:

- $ap_{T1} = A \rightarrow CID \rightarrow IC$
- $ap_{T2} = A \rightarrow CID \rightarrow Parrot$
- $ap_{T3} = A \rightarrow CID \rightarrow GW \rightarrow ABS$
- $ap_{T4} = A \rightarrow CID \rightarrow GW \rightarrow ESP$
- $ap_{T5} = A \rightarrow CID \rightarrow GW \rightarrow LIGHT$
- $ap_{T6} = A \rightarrow CID \rightarrow GW \rightarrow TRUNK$

Using the metric outlined in Section 3.4, we compute the likelihood of a successful attack, the consequences of an attack, and the overall risk of an attack at various levels — including vulnerability, component, ECU, attack path, and system levels.

$$\begin{aligned} prob_{CID} &= prob_{v_{T1}} \times prob_{v_{T2}} \\ &= 0.46 \times 0.72 = 0.33 \\ prob_{IC} &= prob_{Parrot} = prob_{v_{T3}} = 0.72 \\ prob_{GW} &= prob_{v_{T4}} = 0.33 \\ prob_{LIGHT} &= prob_{TRUNK} = prob_{v_{T5}} = 0.59 \\ prob_{ABS} &= prob_{ESP} = prob_{v_{T5}} = 0.59 \end{aligned}$$

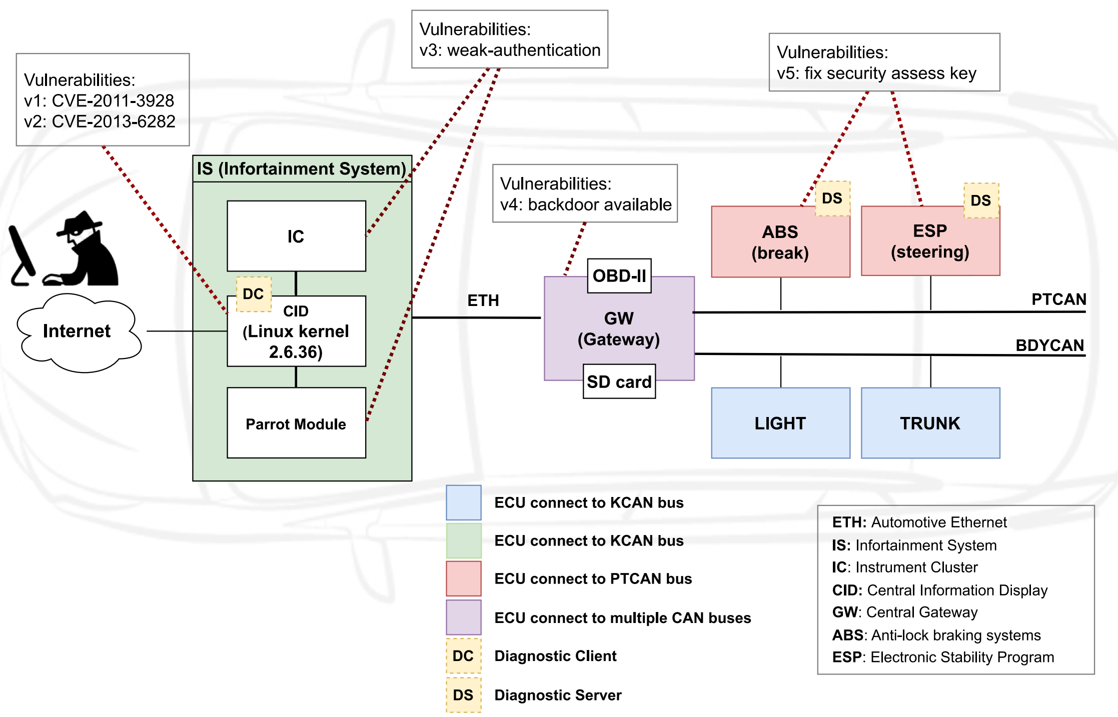


Fig. 5. In-vehicle network architecture of a Tesla Model S vehicle.

Table 5
Vulnerabilities information within the Tesla Model S vehicle.

Notations	CVE-ID	Components	Description	CVSS 3.1
v_{T1}	CVE-2011-3928	CID	The browser runs an old version of AppleWebKit/535.17. It exploits memory corruption (CWE362) in the Webkit engine (libwebkit-hbas-NBT.so) to gain a shell with browser privilege	Base score: 7.1 CIA impact: 2.47, likelihood: 0.72 AV:N/AC:L/PR:N/UI:R/S:C/C:L/I:L/A:L
v_{T2}	CVE-2013-6282	CID	The Linux kernel lack of user input validation which leads to bash command with root privilege in CID	Base score: 8.6 CIA impact: 4, likelihood: 0.46 AV:L/AC:L/PR:N/UI:R/S:C/C:H/I:H/A:H
v_{T3}	No CVE-ID	IC and Parrot	Weak authentication on IC and Parrot allows an attacker to get root access without a password	Base score: 8.8 CIA impact: 3.53, likelihood: 0.72 AV:A/AC:L/PR:N/UI:N/S:C/C:L/I:H/A:L
v_{T4}	No CVE-ID	Gateway	Exist of a backdoor that opens a shell entry with fixed security token in firmware	Base score: 8.0 CIA impact: 4, likelihood: 0.33 AV:A/AC:H/PR:L/UI:N/S:C/C:H/I:H/A:H
v_{T5}	No CVE-ID	ECU with diagnostic server	The Security Access service of the UDS protocol uses weak algorithms. This enables an attacker to incorporate diagnostic functions and manipulate updates on the ECU.	Base score: 9.0 CIA impact: 4, likelihood: 0.59 AV:A/AC:L/PR:L/UI:N/S:C/C:H/I:H/A:H

In the process of evaluating the consequences of a potential attack, both security and safety are given equal weight in our calculations, with both w_{cia} and w_s set to 1. As per the safety criteria outlined in Table 3, we assign a safety impact score of 1 to CID, IC, Parrot, TRUNK, and LIGHT, as a compromise of these components could result in minor injuries due to changes in vehicle lighting or inaccuracies in the infotainment system. The gateway (GW) is given a value of 2 because if it becomes inaccessible, it could lead to serious harm. The safety impact rating for ABS and ESP is 3 since any compromise affecting steering and throttle could result in life-threatening injuries to those in the vehicle or nearby pedestrians.

$$
\begin{aligned}
imp_{CID} &= w_{cia} \times impcia_{CID} + w_s \times imps_{CID} \\
&= 1 \times 4 + 1 \times 1 = 5 \\
imp_{IC} &= imp_{Parrot} \\
&= w_{cia} \times impcia_{IC} + w_s \times imps_{IC} \\
&= 1 \times 3.53 + 1 \times 1 = 4.53 \\
imp_{GW} &= w_{cia} \times impcia_{GW} + w_s \times imps_{GW} \\
&= 1 \times 4 + 1 \times 2 = 6 \\
imp_{LIGHT} &= imp_{TRUNK} \\
&= w_{cia} \times impcia_{LIGHT} + w_s \times imps_{LIGHT} \\
&= 1 \times 3.93 + 1 \times 1 = 4.93 \\
imp_{ABS} &= imp_{ESP} \\
&= w_{cia} \times impcia_{ABS} + w_s \times imps_{ABS} \\
&= 4 + 3 = 7
\end{aligned}
$$

Next, we calculate the risk of each component using Eq. (13), which considers both the probability of a successful attack and its impact.

$$
\begin{aligned}
risk_{CID} &= prob_{CID} \times imp_{CID} \\
&= 0.33 \times 5 = 1.65
\end{aligned}
$$

$$
\begin{aligned}
risk_{IC} &= risk_{Parrot} = prob_{IC} \times imp_{IC} \\
&= 0.72 \times 4.53 = 3.26 \\
risk_{GW} &= prob_{IC} \times imp_{IC} \\
&= 0.33 \times 6 = 1.98 \\
risk_{LIGHT} &= risk_{TRUNK} = prob_{LIGHT} \times imp_{LIGHT} \\
&= 0.59 \times 4.93 = 2.91 \\
risk_{ABS} &= risk_{ESP} = prob_{ABS} \times imp_{ABS} \\
&= 0.59 \times 7 = 4.13
\end{aligned}
$$

The risk for each path and the overall system risk is calculated using Eqs. (15) and (16)

$$
\begin{aligned}
risk_{ap_{T1}} &= risk_{CID} + risk_{IC} = 1.65 + 3.26 = 4.91 \\
risk_{ap_{T2}} &= risk_{CID} + risk_{PARROT} = 1.65 + 3.26 = 4.91 \\
risk_{ap_{T3}} &= risk_{CID} + risk_{GW} + risk_{ABS} \\
&= 1.65 + 1.98 + 4.13 = 7.76 \\
risk_{ap_{T4}} &= risk_{CID} + risk_{GW} + risk_{EPS} \\
&= 1.65 + 1.98 + 4.13 = 7.76 \\
risk_{ap_{T5}} &= risk_{CID} + risk_{GW} + risk_{LIGHT} \\
&= 1.65 + 1.98 + 2.91 = 6.54 \\
risk_{ap_{T6}} &= risk_{CID} + risk_{GW} + risk_{TRUNK} \\
&= 1.65 + 1.98 + 2.91 = 6.54 \\
risk_{S} &= risk_{ap_{T1}} + risk_{ap_{T2}} + risk_{ap_{T3}} + risk_{ap_{T4}} \\
&\quad + risk_{ap_{T5}} + risk_{ap_{T6}} \\
&= 4.91 + 4.91 + 7.76 + 7.76 + 6.54 + 6.54 \\
&= 38.42
\end{aligned}
$$

The total risk for this network is quantified as 38.42, which reflects the overall risk level of the system. This value considers every possible attack route that an attacker could leverage to breach the system. By evaluating the risk associated with each possible route of attack, those in charge of security can identify the most dangerous ones and implement suitable measures to reduce the risk or remove the attack path completely.

4.3. Defence model

To minimize security risks in the vehicle system as outlined in the two case studies, we utilize various defence methods on different components. Some methods can eliminate vulnerabilities, while others can only decrease the likelihood of successful exploitation, thus reducing the overall system risk.

DM1 - Patch management: This method involves providing patches for vulnerabilities in HU and TCB ECU. However, not all vulnerabilities can be patched due to certain factors: (1) Vehicles have complex systems, comprising diverse hardware, software, and safety-critical components. Patching one software vulnerability requires thorough testing to ensure it does not introduce new risks and maintains compatibility with other components. (2) The patching process often requires user involvement. In cases where physical access is needed to patch vulnerabilities, it could become costly or delayed. In situations where the vehicle features OTA update capabilities, the owner's consent and participation are necessary, which can also lead to delays. In this defence model, we will analyse how system security changes when patches are applied to all software vulnerabilities versus when patches are applied to a subset of known vulnerabilities.

DM2 - Access Control on ZGM: The central gateway in a vehicle is crucial as all automotive messages must pass through it. It connects different domains within the vehicle, allowing CAN and UDS messages from the HU and TCB to reach the ECUs of various domains. Therefore, implementing countermeasures at the Gateway ZGM can reduce the probability of a successful attack from the HU and TCB reaching other ECUs. This method involves implementing access control in the gateway, as shown in studies (Luo and Hu, 2018; Song et al., 2016). The access control method can mitigate certain types of attacks, including fuzzing and denial of service (DoS).

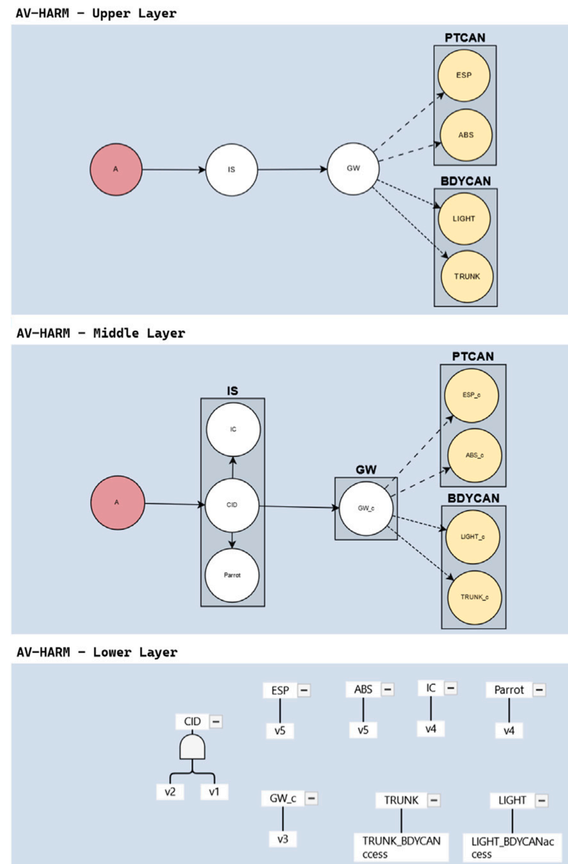


Fig. 6. Attack paths in AV-HARM for the Tesla Model S vehicle.

DM3 - Strong algorithm on UDS server: To reduce the likelihood of successfully sending diagnostic messages to other ECUs via UDS messages, the diagnostic service should apply robust Security Access methods to the UDS server, as highlighted in studies (Thompson, 2022). This mitigation method involves designing more robust cryptographic algorithms for the UDS server.

In the following section, we will conduct an in-depth security evaluation of these various defence models, using two specific case studies: the BMW i3 and the Tesla Model S.

5. Comparative analysis and security evaluation

In this section, we perform a comparative analysis of the system models, attack scenarios, and defence strategies discussed in Sections 4.1 and 4.3, which are limited to the BMW i3 vehicle. This analysis aims to (1) compare the security risks of AV networks under various scenarios using the metrics from the previous section, and (2) assess the effectiveness of the countermeasure methods.

5.1. Security analysis based on different system models

5.1.1. Effect of the number of diagnostic clients on the system security

In the case study scenario in Section 4.1, there are two diagnostic clients in the system: HU-Intel and TCB-Qualcomm. This experiment aims to analyse the impact of reducing the number of diagnostic

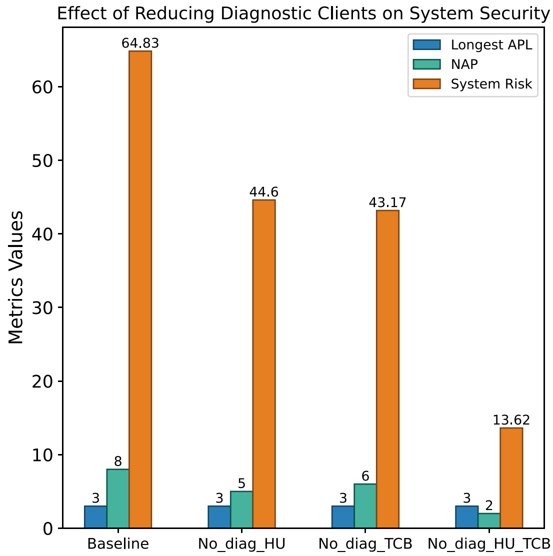


Fig. 7. Effect of reducing diagnostic clients at HU and TCB on Longest APL, NAP and System Risk.

clients on system risk, the number of attack paths (NAP), and attack path length (APL). We evaluate four cases: (1) Baseline scenario with two diagnostic clients (HU-Intel and TCB-Qualcomm); (2) No diagnostic client in HU-Intel (No_diag_HU); (3) No diagnostic client in TCB-Qualcomm (No_diag_TCB); (4) No diagnostic clients in both HU-Intel and TCB-Qualcomm (No_diag_HU_TCB).

The results are shown in Fig. 7. The length of the attack paths remains consistent across all scenarios, suggesting that reducing the number of diagnostic clients does not influence the depth of penetration an attacker can achieve within the system. The attacker still needs to bypass three components to reach Attack Goal 3, which involves controlling the ECU behind the Gateway. However, the number of attack paths and system risk decreases when either the HU-Intel or TCB-Qualcomm diagnostic clients are removed. The removal of the TCB-Qualcomm diagnostic client has a slightly greater positive impact on system risk compared to the HU-Intel, though it does not reduce the number of attack paths. The most significant reduction is observed when both HU-Intel and TCB-Qualcomm are removed. This experiment suggests that the presence of diagnostic clients in the system introduces additional vulnerabilities, increasing both the number of attack paths and the overall system risk. While the diagnostic client is necessary for many features such as remote diagnostic and over-the-air updates, careful management and potential reduction of these diagnostic clients could enhance the security of the automotive system.

5.1.2. Effect of the number of diagnostic servers on the system security

In this experiment, we analyse the impact of the diagnostic server on system risk and the number of attack paths. Specifically, we examine how system security changes with an increase in the number of diagnostic servers in the CAN bus. Modern vehicles typically have up to 100 ECUs (Electronic Control Units) to manage various automotive system functions. However, for our case study, we consider a scenario with only four ECUs: BDC and FLER in the KCAN bus, and EDME and EPS in the PTCAN bus. The KCAN bus is for infotainment and body control functions, while the PTCAN bus connects critical ECUs. We will expand the number of ECUs with diagnostic server functions across these two CAN buses. We assume that an attacker can always connect to the diagnostic server from the diagnostic client due to weak Security Access Service v_{B7} algorithms. We assess five scenarios: (1) No diagnostic server on either CAN bus (No_DS); (2) Baseline scenario with two ECUs on both the KCAN bus and the PTCAN bus (Baseline); (3) Increasing the ECUs on the KCAN bus to 30 (30_KCAN); (4) Increasing the ECUs on the PTCAN bus to 30 (30_PTCAN); and (5) Having 30 ECUs on each CAN bus (30_KCAN_PTCAN).

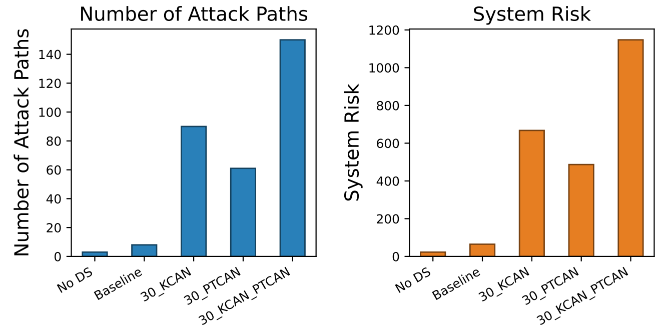


Fig. 8. Effect of the population of the Diagnostic Server on KCAN and PTCAN bus on NAP and system risk.

The results are compared and summarized in Fig. 8. Fig. 8 yields several key insights. First, adding more ECUs with diagnostic servers on the KCAN bus poses a higher risk to the PTCAN bus. This impact on system risk and attack paths is more substantial than similar increases on the PTCAN bus, even when the PTCAN bus connects to critical ECUs. While the plots indicate that KCAN shows a higher risk, this is primarily because the system's risk depends on the number of attack paths. This does not imply that ECUs on the non-critical KCAN bus have a higher inherent risk. Each ECU in KCAN has two distinct attack paths, while PTCAN has only one. However, the plots do not indicate that KCAN's risk is double that of PTCAN, as PTCAN carries a higher overall risk due to specific vulnerabilities and its critical role in the system. The plots show that as the number of access nodes increases, both system risk and attack paths rise proportionately. These results suggest that system security can significantly depend on the number of ECUs with a diagnostic server. Weak security access algorithms in the diagnostic server can create significant security vulnerabilities if an attacker can access the diagnostic client. Therefore, effectively managing and implementing strong encryption algorithms in the diagnostic server is essential for maintaining a secure automotive system.

5.2. Security analysis based on different attack models

5.2.1. Effect of the population of vulnerabilities on the system security

Our goal is to investigate the relationship between the number of known vulnerabilities within the vehicle network, the attack path, and system risks. To this end, we increased the number of vulnerabilities in each component from the baseline scenario to 10. This approach allows us to examine the system risks and attack paths across different scenarios. We employed a script to filter out 706 vulnerabilities with available CVSS 3.0 scores and randomly added 10 vulnerabilities, along with their impact and probability based on CVSS 3.0, into every component. These vulnerabilities were incorporated using OR gate logic.

Fig. 9 displays the results of our analysis. The chart reveals that adding vulnerabilities to different components shows unique risk increase trends. Vulnerabilities in the ZGM cause a sharp increase in system risk because they create a large number of new attack paths compared to the baseline scenario. Similarly, increasing vulnerabilities in HU-Intel exhibit a significant risk increase. Other components, such as the HU-Jacinto, show more gradual risk increases as vulnerabilities are added. This is because, in the baseline scenario, the probability of attack success and the impact of vulnerabilities in HU-Jacinto are already high. Therefore, adding more vulnerabilities does not significantly affect the overall system security risk. The system risk in TCB-Qualcomm remains the same because the probabilities and impact of attack success on TCB-Qualcomm are at their maximum in the baseline scenario. This suggests that the effects of introducing new vulnerabilities can vary greatly depending on the specific component impacted, with certain components, such as the Gateway, posing a much higher risk to system security.

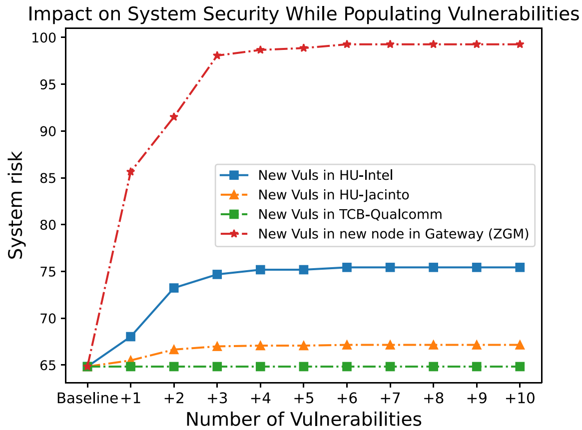


Fig. 9. Effect of the population of the vulnerability on System Risk.

Table 6
K-zero day safety and NAP for different distributions of zero-day vulnerabilities.

Distribution of zero-day vulnerabilities	k-zero day safety	NAP
No vulnerability	–	0
In HU_Intel	1	1
In HU_Jacinto	–	0
In TCB_Qualcomm	1	1
In all node	1	2

5.2.2. Effect of k-zero day safety on the system security

In this experiment, we assume that all the known vulnerabilities have been patched, leaving the system with no known vulnerabilities. We applied Algorithm 1 to determine the number of zero-day vulnerabilities required to compromise a network asset. The higher the value of k, the more secure the system is considered to be. We also used AV-HARM to calculate the number of attack paths (NAP) for each scenario. For the experiment, we made the following assumptions: (1) Each node has only one zero-day vulnerability that can fully compromise the root privileges of the network; (2) The target node is the EPS unit. The results are presented in Table 6. The table shows the k-zero day safety and the NAP for different distributions of zero-day vulnerabilities across the system.

The results indicate that the system's security is not robust. A single zero-day vulnerability with root privilege can compromise the target EPS unit, which is Attack Goal 3, the critical one. This shows that it is relatively easy for an unauthorized user to gain control of the critical system. In this scenario, we focus on two components with the greatest impact on system vulnerability: the HU and TCB. A zero-day vulnerability exists in either of these components, which can allow an attacker to control the EPS unit. This is because both the HU and TCB have built-in diagnostic client functions that allow them to send diagnostic tasks to the ECUs on the critical PTCAN bus. Additionally, introducing zero-day vulnerabilities across all nodes increases the number of attack paths to two. However, the k-zero day safety remains constant at one, showing that while the network becomes more vulnerable to multi-vector attacks, the minimum effort required to compromise the target stays the same.

5.3. Evaluation of different defence methods

This section evaluates the different defence methods outlined in Section 4.3.

5.3.1. Effect of vulnerability patching and rank significant components and vulnerabilities

As patching all vehicle software vulnerabilities may be impossible due to the high cost and dependency on the vehicle owner, we analyse

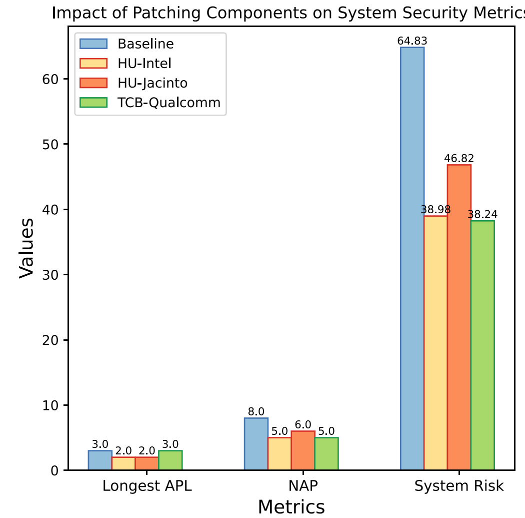


Fig. 10. Effect of patching vulnerabilities in different components on system risk, NAP, and Longest APL.

to compare the effect of patching software vulnerabilities. From this analysis, we aimed to rank the significant components and vulnerabilities, identifying which are most critical and need to be addressed first. To rank the significant components, we patched all vulnerabilities in each component and compared how they affect System Risk, NAP, and Attack Path Length.

Fig. 10 displays the effect of patching vulnerabilities on the metrics Longest APL, NAP, and System Risk for different components. These components include HU-Intel, HU-Jacinto, and TCB-Qualcomm. Prior to applying the defence method, system risk was at its peak, particularly for TCB-Qualcomm. This implies that patching vulnerabilities in TCB-Qualcomm could significantly reduce system risk. Comparing NAP, patching vulnerabilities on HU-Intel and TCB-Qualcomm is more effective, reducing the NAP from 8 to 3. The Longest APL metric shows minor variations among the components, suggesting a more uniform impact across different components.

Based on System Risk, the components can be ranked in order of significance as follows: (1) TCB-Qualcomm, (2) HU-Intel, and (3) HU-Jacinto.

Similar to the previous part, this experiment involves patching software vulnerabilities. However, instead of patching all vulnerabilities in each component, we patched each vulnerability separately to determine which vulnerabilities were the most significant. We only patch the vulnerabilities with determined CVE-ID including v_{B1}, v_{B2}, v_{B3}, v_{B5} and v_{B6}.

Fig. 11 illustrates the impact of patching vulnerabilities on the Longest APL, NAP, and System Risk metrics for various vulnerabilities. These include Baseline, v_{B1}, v_{B2}, v_{B3}, v_{B5} and v_{B6}. The System Risk metric is highest for the Baseline vulnerability and significantly lower for the v_{B5} and v_{B6} vulnerability, suggesting that patching these two vulnerabilities can substantially reduce system risk. The metric of NAP and APL does not change after patching a single vulnerability. This is because each component has multiple vulnerabilities connected with AND/OR logic; therefore, patching a single vulnerability may not reduce the NAP and APL. However, from the results of both system risks, NAP and APL, we can see that ranking vulnerabilities based on system risk is more effective. From the result, we can rank the significant vulnerability based on system risk as follows: (1) v_{B5} and v_{B6}, (2) v_{B3}, (3) v_{B1}, (4) v_{B2}.

In general, these two experiments help identify which components require priority in vulnerability patching and which vulnerabilities are critical to address. This approach effectively mitigates security risks, minimizes the number of attack paths, and manages attack path lengths.

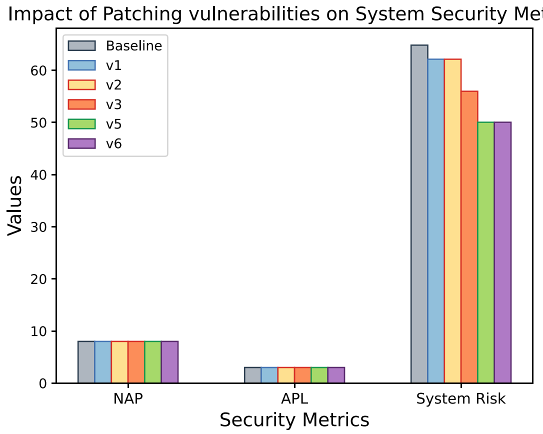


Fig. 11. Effect of patching individual vulnerabilities on NAP, Longest APL, and System risk.

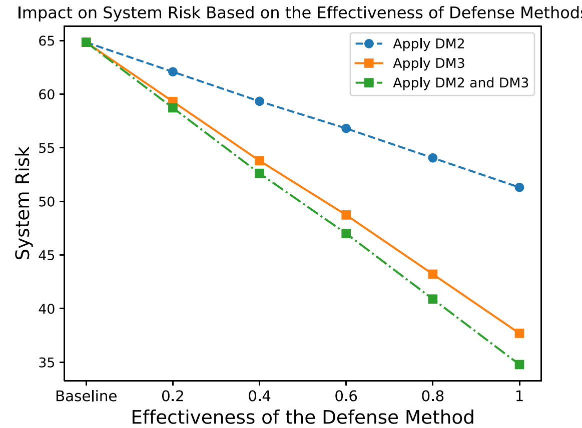


Fig. 12. Effect of applying compensating defence methods on system risk with the increase of effectiveness of the defence method.

5.3.2. Effect of applying compensating defence controls

Not all system vulnerabilities can be fully patched. For instance, vulnerability v_{B7} arises from using weak algorithms in the UDS protocol, allowing an attacker to inject malicious messages into the ECU. Patching alone may not fully eliminate this vulnerability. Therefore, compensating defence controls should be applied. These controls are defence mechanisms targeting specific parts of the system to meet security requirements when it is not feasible to fully remove the vulnerability. Similarly, defence models DM2 and DM3, described in Section 4.3, cannot fully prevent attacks exploiting v_{B7}, but they can reduce the probability of a successful attack. Using ACT and Eq. (1), we aim to perform a sensitivity analysis of the effectiveness of mitigation methods DM2 and DM3 in this experiment. We assume that attacks can always be detected, meaning the detection probability $prob_{dt}$ is always 1. For each defence method, we vary the effectiveness of the defences from 0.2 to 1.0 to observe their impact on overall system risk.

In this experiment, we use ACT and Eq. (1) to perform a sensitivity analysis of the effectiveness of mitigation methods DM2 and DM3. We operate under the assumption that attacks can always be detected, meaning the detection probability $prob_{dt}$ is consistently 1. For each defence method, we progressively alter the effectiveness of the defences from 0.2 to 1.0 to observe their influence on the overall system risk. For DM2, which applies access control to the ZGM, the ZGM will detect CAN messages with incorrect CAN identification. From the HU-Jacinto, an attacker can directly send a CAN message to control the BDC and FLER, as these ECUs are on the same CAN bus as the HU. Therefore, we will adjust the probability of a successful attack on these ECUs based on the effectiveness of the defences, ranging from 0.2 to 1.0.

DM3 involves using a strong algorithm on the UDS server. Since the UDS server is integrated into the BDC, EDME, and EPS, applying DM3 will affect the attack success probability for these ECUs. If the defence method's effectiveness is 0 (baseline scenario), it means there is no defence for the component. Conversely, if the effectiveness is 1, it means the component is completely unreachable. Fig. 12 shows the result of that experiment. The system risk decreases linearly as the effectiveness of the defence methods increases. Also, applying DM3 is more effective than DM2 in reducing the system risk when applied individually.

6. Limitations and future work

In this paper, we introduce AV-HARM, a graph-based security model designed to capture attack paths and evaluate security measures in autonomous vehicles. The model's effectiveness has been demonstrated through use cases and comparative analyses. However, some areas require further improvement and exploration.

- **Integration of Security and Safety Impacts:** AV-HARM currently considers both the security and safety impacts of a vehicle, treating these two attributes as independent. However, in reality, the relationship between security and safety is often interdependent, as a single software vulnerability can simultaneously affect both aspects. Our current model does not yet address the interplay between these two impacts. Future work will focus on integrating the safety attribute with security impacts, particularly under the simulation of cyber-attacks, to provide a more comprehensive analysis.
- **Extension to Multiple Vehicles:** At present, AV-HARM is designed to analyse the security of a single vehicle. However, autonomous vehicles often have components that enable communication between AVs. We have not yet considered the security implications of interactions between multiple vehicles and how the security compromise of one vehicle might impact others. Future work will extend the model to analyse the security and safety impacts across multiple vehicles during a cyber-attack.
- **Dynamic Attacker Behaviour:** Our current model does not account for attackers' dynamic behaviours. Future research could explore reinforcement learning approaches to model these behaviours. This would enable an analysis of attackers' evolving strategies and decision-making processes, such as their responses to defence mechanisms or exploitation of various system vulnerabilities. By incorporating a reinforcement learning component, the model could simulate more realistic, dynamic attack paths, which potentially improve the mitigation of security risks in AV systems.
- **Consideration of Vehicle Movement and Attack Timing:** AV-HARM does not currently account for the dynamics of vehicle movement and the timing of attacks, which can significantly influence the likelihood of successful attacks. The model presently considers the attacker's location relative to the vehicle's system, not the vehicle itself. In our future work, we will incorporate factors such as the moving state of the vehicle and the attack timing. For example, the proximity between a vehicle and an attacker can critically affect the probability of success of certain types of attacks, such as short-range attacks on Cellular, Wi-Fi, and Bluetooth systems. Attack timing is also crucial, as it determines when an attack can impact the vehicle's safety. For instance, the timing of an infotainment system exploit may differ from the timing needed for injecting automotive messages to control the vehicle physically.
- **Validation and Verification:** We used the actual vehicle architecture of a BMW and a Tesla car, along with real-world

vulnerabilities and attack methods, to validate our model. However, further verification through simulations or test beds is necessary to enhance the reliability and robustness of AV-HARM. Future work will involve implementing these additional validation steps to ensure the model's effectiveness and accuracy in diverse scenarios.

7. Conclusions

AV systems are susceptible to various types of cyber-attacks, such as network attacks, protocol attacks, and sensor-based attacks. Current defence methods often struggle to meet industry criteria, making security modelling an effective proactive approach for identifying potential attack vectors and implementing effective security protections. Despite this, there are gaps in the existing literature on AV security modelling, including (1) the lack of incorporation of real-world vulnerabilities in literature-based analyses, (2) no previous work has evaluated the risk of zero-day vulnerabilities in AV networks, and (3) a lack of clear methods and metrics for evaluating models.

In this paper, we have introduced a novel graphical model named AV-HARM to analyse the cybersecurity of AV systems. This model effectively identifies potential attack paths and associated risks within integrated AV systems. We have outlined a comprehensive framework for AV system security analysis, including formal definitions of both the system and the model. Additionally, We have proposed specific metrics to quantify the security and safety of AV systems at different levels, considering both known vulnerabilities and potential future threats.

Our model integrates in-vehicle network architecture with a threat model for each vulnerable component, helping decision-makers identify weak points and apply suitable mitigation strategies. We have illustrated the practical applications of AV-HARM through two case studies, BMW and Tesla, showcasing its relevance to real-world scenarios. These case studies cover a range of physical and remote attacks, including USB-based attacks, cellular networks, software vulnerabilities, OS kernel exploits, and the injection of automotive messages to control the vehicle, demonstrating the model's versatility in addressing diverse vehicle architectures and attack vectors.

Additionally, we have built a web application that automatically visualizes the attack model and calculates security metrics. Furthermore, we have conducted comparative analyses on different system models, attack models, and defence models to evaluate the risk in each case and the effectiveness of varying defence methods. Our results can assist security decision-makers in identifying the most vulnerable components of AV systems and evaluating the effectiveness of different defence methods that can be implemented.

CRediT authorship contribution statement

Nhung H. Nguyen: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Mengmeng Ge:** Writing – review & editing, Supervision, Methodology, Formal analysis, Conceptualization. **Jin-Hee Cho:** Writing – review & editing, Supervision, Methodology, Conceptualization. **Terrence J. Moore:** Writing – review & editing, Supervision, Methodology, Conceptualization. **Seunghyun Yoon:** Writing – review & editing, Supervision, Methodology, Conceptualization. **Hyuk Lim:** Writing – review & editing, Supervision, Methodology, Conceptualization. **Frederica Nelson:** Writing – review & editing, Supervision, Methodology, Conceptualization. **Guangdong Bai:** Writing – review & editing, Validation, Supervision, Methodology, Conceptualization. **Dan Dongseong Kim:** Writing – review & editing, Validation, Supervision, Project administration, Methodology, Formal analysis, Conceptualization.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Nhung H. Nguyen reports financial support was provided by Vingroup Science & Technology Scholarship Program. Dan Dongseong Kim reports financial support was provided by International Technology Center Indo-Pacific ITC IPAC. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This study was sponsored by the Vingroup Science and Technology Scholarship Program for Overseas Study for Master's and Doctoral Degrees.

This material is based upon work supported by the International Technology Center Indo-Pacific (ITC IPAC) under Contract No. FA520924C0027.

Data availability

Data will be made available on request.

References

Behfarnia, A., Eslami, A., 2018. Risk assessment of autonomous vehicles using Bayesian defense graphs. In: IEEE Vehicular Technology Conference. Vol. 2018-Augus, http://dx.doi.org/10.1109/VTCFall.2018.8690732, arXiv:1903.02034.

BMW, 2014. BMW Technical Training. Technical Report, BMW, URL: https://www.leschroniquesdegoliath.com/wp-content/uploads/2018/01/Mercedes-05_I01-General-Vehicle-ElectronicsGB.pdf.

Cai, Zhiqiang, Wang, Aohui, Zhang, Wenkai, Gruffke, M., Schweppe, H., 2019. 0-days and Mitigations: Roadways to exploit and secure connected BMW cars. In: Black Hat USA. p. 39.

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: Proceedings of the 20th USENIX Security Symposium. pp. 77–92.

Enoch, Simon Yusuf, Lee, Jang Se, Kim, Dong Seong, 2021. Novel security models, metrics and security assessment for maritime vessel networks. Comput. Netw. (October 2020), http://dx.doi.org/10.1016/j.comnet.2021.107934.

ETSI Technical Committee ITS, 2017. TR 102 893 - V1.2.1: Intelligent Transport Systems (ITS); Security; Threat, Vulnerability, and Risk Analysis (TVRA). Technical Report, European Telecommunications Standards Institute, pp. 1–88.

FIRST.org Inc., 2019. Common Vulnerability Scoring System v3.1 Specification Document. Technical Report, FIRST, pp. 1–21, URL: https://www.first.org/cvss/v3.0/cvss-v30-specification_v1.9.pdf.

Ge, Mengmeng, Hong, Jin B., Guttmann, Walter, Kim, Dong Seong, 2017. A framework for automating security analysis of the internet of things. J. Netw. Comput. Appl. 83 (April 2016), 12–27. http://dx.doi.org/10.1016/j.jnca.2017.01.033.

Hong, Jin, Kim, Dong Seong, 2012. HARMs: Hierarchical attack representation models for network security analysis. In: Proceedings of the 10th Australian Information Security Management Conference. AISM 2012, pp. 74–81. http://dx.doi.org/10.4225/75/57b559a3cd8da.

Hoppe, T., Kiltz, S., Dittmann, J., 2011. Security threats to automotive CAN networksPractical examples and selected short-term countermeasures. Reliab. Eng. Syst. Saf. 96 (1), 11–25. http://dx.doi.org/10.1016/j.ress.2010.06.026.

International Organisation for Standardisation (ISO), 2018. BSI Standards Publication: Road Vehicles — Functional Safety (ISO 26262-3). Standard ISO 26262-3, International Organisation for Standardisation.

Jungebloud, Tino, H. Nguyen, Nhung, Seong Kim, Dong, Zimmermann, Armin, 2024. Hierarchical model-based cybersecurity risk assessment during system design. In: ICT Systems Security and Privacy Protection. Springer Nature Switzerland, pp. 30–44. http://dx.doi.org/10.1007/978-3-031-56326-3_3.

Karray, Khaled, Danger, Jean Luc, Guilley, Sylvain, Abdelaziz Elaabid, M., 2018. Attack tree construction and its application to the connected vehicle. In: Cyber-Physical Systems Security. pp. 175–190. http://dx.doi.org/10.1007/978-3-319-98935-8_9.

Khalid Khan, Shah, Shiwakoti, Nirajan, Stasinopoulos, Peter, 2022. A conceptual system dynamics model for cybersecurity assessment of connected and autonomous vehicles. Accid. Anal. Prev. 165, 106515. http://dx.doi.org/10.1016/j.aap.2021.106515.

Kohnfelder, L., Garg, P., 1999. The Threats to Our Products. Vol. 33, Microsoft Interface, Microsoft Corporation.

Koscher, K., Czeskis, A., Roesner, F., Patel, S., Kohno, T., Checkoway, S., McCoy, D., Kantor, B., Anderson, D., Snachm, H., Savage, S., 2010. Experimental security analysis of a modern automobile. In: Proceedings - IEEE Symposium on Security and Privacy. IEEE, pp. 447–462. http://dx.doi.org/10.1109/SP.2010.34.

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. In: CSCS '21, Association for Computing Machinery, New York, NY, USA, http://dx.doi.org/10.1145/3488904.3493378.

Luo, F., Hu, Q., 2018. Security Mechanisms Design for In-Vehicle Network Gateway. SAE Technical Papers, Vol. 2018-April, pp. 1–8. http://dx.doi.org/10.4271/2018-01-0018.

Luo, Feng, Jiang, Yifan, Zhang, Zhaojing, Ren, Yi, Hou, Shuo, 2021. Threat analysis and risk assessment for connected vehicles: A survey. Secur. Commun. Netw. 2021, http://dx.doi.org/10.1155/2021/1263820.

Miller, C., 2019. Lessons learned from hacking a car. IEEE Des. Test 36 (6), 7–9. http://dx.doi.org/10.1109/MDAT.2018.2863106.

Miller, C., Valasek, C., 2015. Remote exploitation of an unaltered passenger vehicle. In: Defcon 23. Vol. 2015, pp. 1–91, URL: http://illmatics.com/RemoteCarHacking.pdf.

Monteuuis, J.P., Boudguiga, A., Zhang, J., Labiod, H., Servel, A., Urien, P., 2018. SarA: Security automotive risk analysis method. In: CPSS 2018 - Proceedings of the 4th ACM Workshop on Cyber-Physical System Security, Co-Located with ASIA CCS 2018. pp. 3–14. http://dx.doi.org/10.1145/3198458.3198465.

Nie, S., Liu, L., Du, Y., 2017. Free-fall: hacking tesla from wireless to can bus. In: Defcon. pp. 1–16.

Nie, S., Liu, L., Du, Y., Zhang, W., 2018. Over-the-air: How we remotely compromised the gateway, BCM, and autopilot ECUs of TESLA cars. In: BlackHat USA 2018. Vol. 1, pp. 1–19.

Petho, Zsombor, Khan, Intiyaz, Torok, Arpad, 2021. Analysis of security vulnerability levels of in-vehicle network topologies applying graph representations. J. Electron. Test.: Theory Appl. (JETTA) 37 (5–6), 613–621. http://dx.doi.org/10.1007/s10836-021-05973-x.

Roy, Arpan, Kim, Dong Seong, Trivedi, Kishor S., 2010. Poster abstract: ACT: Attack countermeasure trees for information assurance analysis. In: Proceedings - IEEE INFOCOM. IEEE, pp. 4–5. http://dx.doi.org/10.1109/INFCOMW.2010.5466633.

Ruddle, Alastair, Weyl, Benjamin, Idrees, Sajid, Roudier, Y., Friedewald, Michael, Leimbach, Timo, Fuchs, A., Gürgens, S., Henninger, O., Rieke, Roland, Ritscher, M., Broberg, H., Apvrille, L., Pacalet, R., Pedroza, Gabriel, 2009. Security Requirements for Automotive On-Board Networks Based on Dark-Side Scenarios. Deliverable D2.3: EVITA. E-Safety Vehicle Intrusion Protected Applications. Fraunhofer ISI.

SAE International, 2021. SAE J3016: Levels of Driving Automation. Technical Report, SAE International, URL: https://www.sae.org/standards/content/j3016_202104/.

Song, Hyun Min, Kim, Ha Rang, Kim, Huy Kang, 2016. Intrusion detection system based on the analysis of time intervals of CAN messages for in-vehicle network. In: 2016 International Conference on Information Networking. ICOIN, IEEE, pp. 63–68.

Tencent Keen Security Lab, 2016. Gateway Internals of Tesla Motors V6. Technical Report, Tencent Keen Security Lab, URL: https://www.yumpu.com/en/document/view/56589245/gateway-internals-of-tesla-motors.

Tencent Keen Security Lab, 2019. Experimental Security Research of Tesla Autopilot. Technical Report, Tencent Keen Security Lab, p. 38, URL: https://keenlab.tencent.com/en/whitepapers/Experimental_Security_Research/_of_Tesla_Autopilot.pdf.

Thompson, M., 2022. UDS Security Access for Constrained ECUs. SAE International, http://dx.doi.org/10.4271/2022-01-0132.

Valasek, C., Miller, C., 2013. Adventures in Automotive Networks and Control Units. Technical White Paper, p. 99, URL: http://www.ioactive.com/pdfs/IOActive_Adventures_in_Automotive/_Networks_and_Control_Units.pdf.

Wang, L., Jajodia, S., Singhal, A., Cheng, P., Noel, S., 2014. K-zero day safety: A network security metric for measuring the risk of unknown vulnerabilities. IEEE Trans. Dependable Secure Comput. 11 (1), 30–44. http://dx.doi.org/10.1109/TDSC.2013.24.

Nhung H. Nguyen received her B.S. degree from Hanoi University of Science and Technology in Vietnam in 2012. Following her graduation, she spent the next ten years working in the Department of Cyber Security at the Ministry of Public Security, Vietnam, where she focused on cybersecurity systems and tackling cybercrimes. Now, she is a third-year Ph.D. student at the University of Queensland in Brisbane, Australia. Her research topic centres on developing security and safety models and simulations for autonomous vehicles. Her research interests include Autonomous Vehicle Security, System Security, and Graphical Security Modelling.

Mengmeng Ge is a Senior Lecturer in Cybersecurity at the University of Canterbury, New Zealand. Prior to the University of Canterbury, she was a Lecturer in Cybersecurity at RMIT University, Melbourne, Australia and a Lecturer in Cyber and Networking at Deakin University, Geelong, Australia. She received her Ph.D. degree in Computer Science from the University of Canterbury, New Zealand, under the topic of graphical security modelling and assessment for the Internet of Things. Her research interests include Evaluation-based Modelling, the Internet of Things, Moving Target Defence, Cyber Deception, and Intrusion Detection.

Jin-Hee Cho (Senior Member, IEEE) received the M.S. and Ph.D. degrees in computer science from Virginia Tech in 2004 and 2008, respectively, where she has been an Associate Professor with the Department of Computer Science since August 2018 and the Director of the Trustworthy Cyberspace Lab. Prior to joining Virginia Tech, she worked as a Computer Scientist with the U.S. Army Research Laboratory, Adelphi, MD, USA, since 2009. She has published over 120 peer-reviewed technical papers in leading journals and conferences in the areas of trust management, cybersecurity, metrics and measurements, network performance analysis, resource allocation, agent-based modelling, uncertainty reasoning and analysis, information fusion/credibility, and social network analysis. She received the best paper awards in IEEE TrustCom'2009, BRIMS'2013, IEEE GLOBECOM'2017, 2017 ARL's publication award, and IEEE CogSima 2018. She won the 2015 IEEE Communications Society William R. Bennett Prize in the Field of Communications Networking. In 2016, she was selected for the 2013 Presidential Early Career Award for Scientists and Engineers, which is the highest honour bestowed by the U.S. government on outstanding scientists and engineers in the early stages of their independent research careers. She is a member of ACM.

Terrence J. Moore (Member, IEEE) received the B.S. and M.A. degrees in mathematics from American University in 1998 and 2000, respectively, and the Ph.D. degree in mathematics from the University of Maryland, College Park, in 2010. He is currently a Researcher with the Network Science Division, U.S. Army Research Laboratory. His research interests include sampling theory, constrained statistical inference, stochastic optimization, network security, geometric and topological applications in networks, and network science.

Seunghyun Yoon (Member, IEEE) received a B.S. degree in computer science from Handong Global University, Pohang, Republic of Korea, in 2016 and a Ph.D. degree in electrical engineering and computer science from the Gwangju Institute of Science and Technology (GIST), Gwangju, Republic of Korea, in 2021. He has been an Assistant Professor with the Department of Energy Engineering, Korea Institute of Energy Technology (KENTECH), since August 2022. From 2019 to 2020, he was a Visiting Scholar with the Department of Computer Science at Virginia Tech. From 2021 to 2022, he was a Senior Researcher with the Korea Institute of Industrial Technology (KITECH). His research interests include artificial intelligence (AI), software-defined networking (SDN), network resource allocation and optimization, cybersecurity, and moving target defence (MTD). He is also conducting active research on continual learning and federated/distributed learning in AI.

Hyuk Lim (Member, IEEE) received the B.S., M.S., and Ph.D. degrees from the School of Electrical Engineering and Computer Science, Seoul National University, Seoul, Republic of Korea, in 1996, 1998, and 2003, respectively. From 2003 to 2006, he was a Postdoctoral Research Associate with the Department of Computer Science, University of Illinois at Urbana–Champaign, Champaign, IL, USA. From 2006 to 2021, he was a professor at the AI Graduate School and jointly with the School of Electrical Engineering and Computer Science, Gwangju Institute of Science and Technology (GIST), Gwangju, Republic of Korea. He was the Dean of the School of Electrical Engineering and Computer Science and the Director of the GIST Institute for AI. In 2022, he joined the Korea Institute of Energy Technology (KENTECH), Naju, Republic of Korea, as a Full Professor. His research interests include artificial intelligence, cybersecurity, big data privacy, software-defined networking, and wireless communication systems.

Frederica Free-Nelson is a Researcher and the Program Lead with the U.S. Army Research Laboratory (ARL), Adelphi, MD, USA, where she leads research on machine learning and intrusion detection methods and techniques to promote cyber resilience and foster research on autonomous active cyber defence. She manages and negotiates the Research and Project Agreements for ARL between the network security branch and Academia or International Organizations. She is the lead for the Robust low-level cyber-attack resilience for Military Defense (ROLLCAGE) program working in collaboration with the Army Tank Automotive Research, Development and Engineering Center, Office of Naval Research, and Air Force Research Laboratory to build a cohesive invehicular resilient system for defence against sophisticated enemy malware that strives to blend in with normal system activities. She has over 20 years of combined experience in Cybersecurity Research, Software Engineering, and Program Management within the DoD and other federal services, including the Federal Bureau of Investigation and the Department of Justice. She has expertise in successfully leading projects from conception to execution and delivery/transfer. She currently serves as the Chairperson to the International Science Technology (IST-163) Panel – NATO Science & Technology Organization on the topic of Deep Machine Learning for military cyber defence. She participates in the Army Education Outreach Program as an ambassador and a virtual judge for the eCybermission program.

Guangdong Bai is an Associate Professor in the School of Electrical Engineering and Computer Science at the University of Queensland (UQ), Brisbane, Australia. He received his Ph.D. degree from the National University of Singapore and his M.S. and B.S. degrees from Peking University, China. He leads the TrustLab at UQ Cyber Security and Software Engineering and is an Associate Editor of IEEE Transactions on

Dependable and Secure Computing. His research interests include Security and Privacy, Software Engineering, and Formal Methods.

Dong Seong Kim (Senior Member, IEEE) received the Ph.D. degree from Korea Aerospace University in February 2008. He has been an Associate Professor of Cybersecurity at The University of Queensland, Australia, since January 2019. He was a Senior Lecturer/Lecturer of Cybersecurity at the University of Canterbury from August 2011 to December 2018. He was a visiting scholar at the University of Maryland, College Park, in 2007. From June 2008 to July 2011, he was a Postdoctoral Researcher at Duke University. His research interests are in automated cybersecurity modelling and analysis for the Internet of Things, cloud computing, and moving target defence. He was the General Co-Chair of ACISP2019 and the General Chair of IEEE PRDC 2017. He served as a Program Co-Chair of IEEE TrustCom2019, IEEE ICIOT2019, ATIS2017, GraMsec2015, and IEEE DASC2015 and Program Committee Member of international conferences, including IFIP/IEEE DSN, ISSRE, SRDS, and ICC CISS, respectively.