Proceedings of the of the 17th Conference on Computer Science and Intelligence Systems pp. 653–662
DOI: 10.15439/2022F136
ISSN 2300-5963 ACSIS, Vol. 30

Heuristic Risk Treatment for ISO/SAE 21434 Development Projects

Christine Jakobs, Matthias Werner
TU Chemnitz
Chemnitz, Germany
christine.jakobs@informatik.tu-chemnitz.de
matthias.werner@informatik.tu-chemnitz.de

Karsten Schmidt, Gerhard Hansch
AUDI AG
Ingolstadt, Germany
karsten.schmidt@audi.de
gerhard.hansch@audi.de

***Abstract*—Due to new technologies for connectivity, automotive systems shift from a closed to an open system approach. Therefore, automotive systems have a rising demand for security, letting security be an upcoming field in research and practice. Also, the newly published process standard ISO/SAE 21434 demands adjustments in the development process to address cybersecurity. The unique characteristics of automotive systems leave many approaches from other system types inapplicable. This work concentrates on the risk treatment step in the cybersecurity development process. Due to the vast amount of differing terminology, we see the need to define a flexible taxonomy adaptable to several system types and used in systems with normative references. We use this taxonomy to develop a heuristic approach for risk treatment based on a distinct terminology for security requirements. The presented method is extendable to include several trade-off points.**

I. Introduction

With rising interest in connectivity and Car2X, also automotive security gets into focus. ISO/SAE 21434 [1], the process standard for automotive security, defines the security analysis and risk treatment process into three steps. The process starts with a security relevance evaluation [2] for deriving the first criticality and filtering the targets of evaluation to relevant ones. After that, the risk analysis, followed by the risk treatment step, occurs.

Risk analysis is done on the decomposed system and tells a story about five questions. The question about what can go wrong determines possible damage scenarios, weighted according to their damage potential - how bad is this? The threat identification and attack path analysis show how damage scenarios happen. A defined attacker model answers the question about who can do that, which allows deriving the attack probability. The combination of both impacts (damage potential and required attack potential) determines the risk value and enables a criticality ranking.

Risk treatment uses the impacts and risk numbers to consolidate the story of the different risk analyses in the system. In this step, weighting the impacts allows a frine-granular prioritization of the risks. Defense method assignment leverage the impacts and thereby reduces the risks.

The first arising problem when looking for demands and approaches for risk treatment is the vast amount of different terminology. Unclear terminology makes it difficult to understand demands and compare approaches. An example is UNECE No. R155 [3]. While its main section uses the term mitigation, the Annex uses security control, measure, and mitigation without proper definition. The same applies to other sources in the literature. Therefore, we see a need to find a distinct definition of the terms used for the risk treatment in automotive systems.

On the other hand, efficient risk treatment demands a structured data basis of the used mitigations. We ground our method on a simple taxonomy based on a distinct terminology with the possibility of transferring it to other system types. The developed defense method catalog is a cross-product of the taxonomy whereby the taxonomy is independent of concrete system information. The presented defense method catalog covers such system-type-related information.

Literature frequently covers risk treatment theoretically, but an efficient and flexible method is missing. Especially methods transferable to different input sources are rare. Most approaches are for particular systems, e.g., service-oriented and web-based systems. Others are implemented in a framework, demanding unique kinds of data sources. We aim for a flexible heuristic approach for risk treatment. Our approach uses general information from risk analysis rather than detailed system models, which makes the approach adaptable to practical settings.

Security requirement demands from system external sources, like normative or organizational policies, are often mixed with requirements related to specifics of the target of evaluation. After risk treatment, applied trade-offs must adhere to a system's external security demands. Otherwise, the system might be cost-effective but infeasible from a legal point of view. Therefore, we observe the origin of the requirements to enable traceability of the security demands.

Risk analysis determines threats to the system based on attack paths. Those have different configurations. Single threats (one-element attack paths) can be compared to single-points-of-failure in reliability or are preparation attacks for other paths. Prioritizing them in the treatment process first mitigates crucial threats (SPOF) and preparation attacks. The latter directly cut attack paths, reducing the effort of treating those. Updating the risks by determining the impact of the defense method on other risks raises the by-catch and reduces the

overall effort of implementing defense methods.

Risk treatment aims to minimize the risk of system threats by either reducing the impact or raising the required attack potential (reducing the attack probability). It is reasonable to prioritize the impacts in the treatment procedure for the efficiency of the process. Those priorities allow trade-offs between different impact categories or attacker model categories in a cost-pressure situation. Our approach aims for a flexible prioritization of the risk treatment procedure and the risk impact categories to allow different trade-off points.

Based on the relation to the automotive industry, the presented work starts with a short introduction of the normative references for risk treatment in automotive development (Section II). In Section III we introduce our taxonomy for security requirements. Section IV covers the approach for risk treatment. Please note that NDA reasons prevent an exhaustive evaluation of our approach. We tried to include several examples in the text, where appropriate. Section V discusses the limitations and possible further trade-offs for our risk treatment method. In Section VI we conclude our contribution and give an outlook for future work.

II. Normative Reference

Two normative references are relevant for type approval in automotive systems: UNECE No. R155 [3] and ISO/SAE 21434 [1]. The demands of both norms have a different layer of detail.

A. UNECE No. R155

According to UNECE No. R155 [3] the OEM has to protect the vehicle type and to implement "all mitigations [...] which are relevant for the risks identified." [3].

Besides the general demand for mitigation implementation, UNECE No. R155 has three concrete demands:

- Intrusion-Detection for the vehicles of a certain type
- A central monitoring facility for new threats and vulnerabilities
- The use of up-to-date cryptographic modules

Annex 5 provides a list of possible risks and appropriate mitigations. The mitigations listed in the Annex are not concrete methods, but rather categories of mitigations, e.g., "The vehicle shall verify the authenticity and integrity of messages it receives" [3]. The OEM may differ from the provided list of mitigations if it is insufficient to mitigate a certain risk.

In conclusion, UNECE No. R155 demands risk mitigations according to the provided categories in the Annex. The general demands are only OEM-related, one regarding the monitoring facility, and two vehicle-related demands. There is no suggestion regarding threat mitigation techniques or methods.

B. ISO/SAE 21434

ISO/SAE 21434 [1] describes the clauses in a triple of input, the requirements and recommendations, and the output. Input is the necessary and optional predecessor work products from other clauses. The requirements define the demands of the ISO and provide possible methods or procedures. Output defines the work products resulting from this clause.

For the risk treatment, ISO/SAE 21434 demands to use the item definition (model of the target of evaluation), the identified attack paths, and the risk values as results from the risk analysis. Optional inputs are cybersecurity specifications (from former development or higher abstraction levels), previous risk treatment decisions, damage scenarios with their impact rating, and attack paths with feasibility ratings.

ISO/SAE 21434 requires the risk treatment for all identified risks by using one or more treatment options. Those options are the classical ones: risk avoidance and reduction, risk-sharing or retaining. The process documentation must record the decision to retain or share risk. Therefore, ISO/SAE 21434 explicitly allows risk acceptance up to a certain level, as long as this threshold and the retained risks are documented.

Like UNECE No. R155, the ISO does not provide possible methods for risk treatment.

III. Security Defense Requirements

Our taxonomy of security requirements (see Figure 1) has three different categories: the origin, the type as well as the defense methods. The objective of the taxonomy of security requirements is the use in the risk treatment step. Therefore, the scope is on those requirements that apply to the vehicle and directly influence its behavior. Process requirements (e.g., audit, testing) and supporting processes (e.g., documentation) are related to the ecosystem in the development process and therefore excluded.

A. Related Work

In [4] the authors define several levels for the selection of defense methods. Their categorization is more detailed than our approach, e.g., to the circuit level. Since, in practice, the risk analysis for components is done based on a selected hardware platform, certain levels for hardware cannot be taken into account anymore. Therefore, we decided on more abstract and less detailed control categories. The basis for their approach to risk treatment is a rich and detailed data model. The data model allows a very comprehensive analysis of dependencies between defense methods. On the other hand, this also demands to use their entire framework for the security process. Otherwise, the demanded input data model may be impossible to acquire.

Pfleeger [5] describes a categorization of defense methods. The main categories are encryption, software, hardware, and physical. Typically, encryption is no stand-alone method but a feature of another defense method, e.g., secure boot, TLS. Physical controls like locks on doors are not applicable for vehicle security. Pfleeger integrates defense methods implemented as separate functions like intrusion detection systems and password checkers into software and hardware levels. We decided to differentiate between the level the control has its impact and the technical and functional control categories. Those include all applicable controls Pfleeger mentions, just in a different categorization.

Also [6] describes the categorization of security requirements into different layers. Those are regarding the point where the information lies: internally, externally, or both. Additionally, Chung differentiates between the development stage of the scope of the methods. We concentrate only on the run-time stage of the system and leave process demands aside. Procedural methods cannot be assigned to the system in the risk treatment step but accompany the complete development process. Therefore, they are in the scope of development process design, which is a separate topic.

The literature review of Akhunzada et al. [7] results in a thematic taxonomy regarding the different layers of software-defined networks. The categorization is very comprehensive but not transferable to other system domains. We aim for a more pragmatic taxonomy easily adaptable to other system types.

[8] presents another taxonomy based on literature research. The scope of this work is restricted to software integrity protection techniques. Different system views build the basis for the categorization: system view, defense view, and attack view. The evaluated literature is mapped onto the views and correlations are evaluated. Nevertheless, besides the different scopes, the granularity of the taxonomy is not helpful for our approach. We aim for a flexible approach to clarify the terminology needed for risk treatment to use it for the defense method assignment.

B. Origin

Security Requirements arise from different sources. Those are general conditions and system-related requirements. The former depict system external sources while the latter arises from the security analysis of the target of evaluation.

a) General conditions: General conditions are related to the vehicle type. They may not directly support accomplishing a security goal, but non-fulfillment provocate a mission failure.

The most critical source for those requirements is regulatory organizations. Norms like UNECE No. R155 [3] and ISO/SAE 21434 [1], but also country-specific regulations like GB/T 40856-2021 [9] demand certain types of security requirements in a varying level of detail. Examples are the intrusion detection system demand of UNECE No. R155. Those regulatory requirements are relevant for the type approval of the vehicle type. Therefore, a non-accomplishment endangers the possibility of selling the vehicle type, at least in certain countries.

Security-in-depth and security-by-design demand to define certain basic security standards on organizational level [10]. Those are also a source for general conditions. A non-accomplishment of the company policies does not lead to the loss of type approval but endangers the company's internal vehicle audit, e.g., during testing. Also, company policies may have a varying level of detail from distinct methods in a certain variation to general demands.

The nature of general conditions is that they have varying levels of detail. They may not explicitly project to specific defense methods but categories of those. Therefore, they may not directly support accomplishing a security goal (e.g., Confidentiality) to a certain degree but demand a defense method that supports this goal. One idea is to formalize them in a logic-based language to define the projections from the policies to their varying level of detail in the defense method catalog.

b) System-related Requirements: Requirements that arise from the target of evaluation are system-related. Those depict the necessity for defense methods against the risks evaluated in the risk analysis.

The system-related requirements are distinct since they base on specific risks according to threats against security goals and impacts regarding their damage and attack potential. Therefore, system-related requirements are refinable until they demand a defined variation of specific defense methods.

C. Types of Security Requirements

The type of security requirements defines the nature of the security requirements. The taxonomy categorizes them into those directly recognizable in the resulting system or not.

a) Measures: Measures are those requirements that are not directly recognizable in the system. They instead depict system design methods that can be non-technical, procedural, or logical methods against violating a security goal. Examples are the prevention of specific information flows or removing unused software libraries instead of applying expensive defense methods. Measures are typically general conditions or system-related requirements applied to the system during the development process. They are not directly used in the risk treatment step but are rules to verify after risk treatment.

b) Controls: Controls are requirements that are directly or indirectly recognizable in the system. Risk treatment refines those requirements till they depict specific defense methods in certain variations. Controls accomplish the means of risk treatment: reduce, detect, or avoid risks entirely. Examples are safeguards on all system levels like access control, Intrusion Detection Systems or secure communication protocols.

The different layers of the controls represent the defense-in-depth onion model: structural, technical, and functional controls. Structural controls are related to the overall system structure, e.g., network segmentation. They answer the question - How does the system structure look to avoid specific threats? Technical controls defend threats from the internal processing or component side. Examples are controls preventing the manipulation of software at rest or during run-time. Functional controls complete the onion model by providing defense on function level. Those controls define the behavior of connections to and within the system, e.g., with communication protocols and access controls.

D. Method categories

The last part of the security requirement taxonomy are the categories of defense methods. Those categories are dependent on the development project and are adjustable for every vehicle type in automotive development projects. This adjustment ensures that the categories are up-to-date and complete.

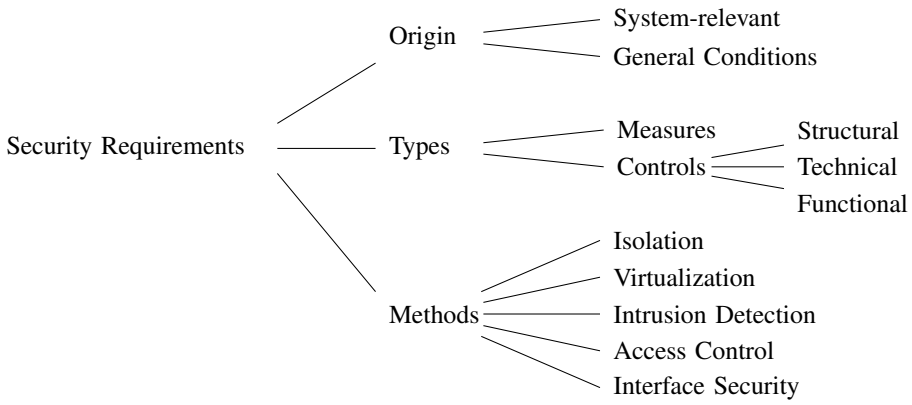

Fig. 1. Taxonomy of security requirements split into origin, types and methods.

In principle, the categories depict classes of defense methods and allow a more straightforward assignment in the risk treatment process. One suggestion is to align the classes with the general conditions, e.g., intrusion detection, isolation, and segmentation. Examples of typical classes for automotive development projects show Figure 1 without a claim for completeness.

E. Catalog of Defense Methods

The presented taxonomy allows a cross-product of types and method categories by assigning concrete defense methods (see Table I).

The assignment clusters the set of available defense methods into the categories from the taxonomy and annotates them with the type of control. Demands from general conditions like state-of-the-art cryptography do not lead to measures assigned to the list of defense methods. Those demands instead limit the scope of the available methods, e.g., non-appropriate cryptographic procedures are not part of the assignment.

Defense methods may be configurable, e.g., secure service-oriented communication (sSOA) with SOME/IP or SOCKS. Those variations can be directly included in the catalog or hidden as variants in the properties of the method. Encryption types are not direct defense methods but used by defense methods, e.g., secure communication protocols. They are therefore variants of the using method.

Defense methods may have no direct impact on the security goals. The impact may be emergent only in combination with other controls, e.g., a hardware security module (HSM) is only helpful in combination with a defense method that uses the cryptographic algorithms and the secure key storage provided by the HSM. Therefore, those methods are not included in the catalog but are variants of the defense method. An example would be TLS in combination with a present HSM [11]. In this case, there would be at least two variants of TLS, with and without an HSM present.

The catalog of defense methods is also subject to change. The catalog needs to be updated and refined in every development cycle, e.g., every vehicle type. Arising new threats may lead to changes during a development cycle. In this case, the responsible security engineers need to be informed about deleted methods and possible substitutions. This accounts also to other system types [10].

TABLE I
CLASSES AND IMPLEMENTATIONS OF DEFENSE METHODS. COLUMNS ARE THE TYPES OF CONTROLS (S=STRUCTURAL, T=TECHNICAL, F=FUNCTIONAL). ROWS INDICATE THE CLASS HIERARCHY AND EXAMPLES FOR CONCRETE IMPLEMENTATIONS. X INDICATES THAT THIS IMPLEMENTATION CAN BE USED FOR THIS CONTROL TYPE.

	Implementations		Control Types S	T	F
Isolation	NW Segmentation				
		VLAN	x		x
		Physical	x		
		Firewall	x	x	x
	Host Segmentation				
		CPU		x	x
	Virtualization				
		SOA Domain			
		Hypervisor			
		Sandboxing		x	x
Intrusion Detection	Runtime				
		IDS	x	x	x
		Logging			x
		Runtime Protection		x	x
	Startup				
		Secure Boot		x	x
		Authenticated Boot		x	x
Interface Security	Communication				
		Secure SOA			x
		TLS			x
		IPsec			x
		SOK			x

F. Properties of Defense Method

Defense methods mitigate threats to security goals. It is possible to derive the threat and damage scenario type from the targeted security goals. Therefore, it is better to assign the security goals as properties of the defense method rather

than the threat and damage scenario types. Following that, the impact on the different security goals is part of the properties. In [5] a range from -2 to +2 for each security goal is suggested. The positive part depicts the advantages of the defense method on security goals, while the negative part illustrates that defense methods may negatively influence certain security goals.

The definition of the effect on the attack potential and types of damage scenarios illustrates a defense method's influence on specific threats. This step reuses the method from the risk analysis, for the attack's potential impact may include the influence on the needed time, knowledge, tools, expertise, and access of the attacker.

The surface, as well as the dependency properties, limit the applicability of defense methods. The surface is regarding the attack surface, which this defense method mitigates. Dependencies relate to supporting processes needed from the technical side for this method (variant) to work or the limited scope of a defense method, e.g., VLAN technology is only applicable for Ethernet LAN.

It is possible to also assign costs for the defense method. Those represent the resource usage in the case of implementing this method. Examples would be processor clocks per byte in the case of DES [12]. In [11] different ways to derive the costs for cryptographic algorithms and examples are provided. Cost definitions allow to include risk treatment directly into resource scheduling procedures [12] which is out of scope in this work.

The different variants of a defense method have their properties assigned. Table II provides an example for securing JTAG. The two variants are disabling JTAG physically, which is the most secure variant but highly influences availability of the system. On the other hand is JTAG security by utilizing cryptography, e.g., SSL [13] which has a compared high effort.

TABLE II
EXAMPLE OF DEFENSE METHOD PROPERTIES FOR SECURING THE JTAG PORT WITHOUT CLAIM FOR COMPLETENESS. THE VALUES FOR COST, IMPACT AND EFFECT ARE VARIABLES TO ASSIGN BASED ON THE GIVEN SETUP.

	Property	**Variants**	
		Physical (Disable)	Cryptographic
	Technology	JTAG	JTAG
	Costs	0	?
Impact	Confidentiality	+2	+1
Impact	Integrity	+2	+1
Impact	Availability	-2	-1
	Surface	Local	
	Dependencies	-	SSL
	Effect	RAP: ?; DP: ?	

IV. RISK TREATMENT

The goal of risk treatment is twofold: Minimizing the risk, which means reducing the costs of non-implementation of defense methods, and minimizing the effort by reducing the costs of implementing defense methods.

In the security requirement taxonomy, general conditions are relevant for the type approval and/or necessary to fulfill OEM demands. Therefore, their costs for not-implementation are infinite. On the other hand, the costs for implementing general conditions are most of the time variable. They are typically related to categories of defense methods. Therefore, the implementation costs depend on the chosen method of the respective category. For system-related requirements, the costs of not-implementation need to be traded against the implementation costs. In the case of low-rated risks, it may be cheaper not to implement costly defense methods.

Also in settings where the direct implementation costs cannot be taken into account, risk treatment needs to take into account those cost-related problems. This can be done by reducing the overall number of defense methods through structured method assignment, e.g., by first cutting attack paths before treating the complete risk. Another possibility is to assign methods which impact several security goals at once instead of using a method per security goal.

The remaining section proceeds through the complete risk treatment process. Because of the already discussed problems with acquiring the costs of defense methods, we count for efficiency through a structured method assignment, and illustrate possible trade-offs and optimization points.

A. Prerequisites

According to ISO/SAE 21434 [1] risk treatment is related to identified risks and attack paths of the target of evaluation. Concrete, those risks whether the impact is against the road user. While it is possible to consider other impact categories, we will concentrate on those risks. The treatment of other risks is subject to by-catch of assigned methods and trade-offs.

The risks, threats, and attack paths, as well as their impact, are analyzed in the risk analysis step. There are different ways to complete this predecessor process step. Prominent are integrated approaches, e.g., MoRA [14] in the automotive world, or approaches developed for different sub-steps of risk analysis, e.g., attack trees [15] [16] [17] for attack path identification and rating.

For the presented approach, it is reasonable to evaluate the risk values in the risk analysis threefold: The risks against the road user, the OEM, and combined. Those three risks allow different options for trade-offs during or after method assignment. According to damage potentials and assumptions are outputs from the risk analysis steps in ISO/SAE 21434. Assumptions may be regarding the importance of specific damage scenarios and threats or related to other items' security process results. Therefore, they must be considered in the risk treatment because of their impact on threats and damage scenarios and traced for later verification steps.

B. Strategy

In general, risk treatment should be done holistically over the complete system. A global approach demands formal constraints for all defense methods and a distinct format for the input data. The latter is a big problem, especially in distributed

development environments like the automotive industry. Also, clearly defined tools for risk analysis allow different levels of abstraction and typically use natural language to formulate the damage scenarios and threats. Therefore, the bigger the input space for the risk treatment, the more significant the divergence between the input data. This divergence is also because the automotive environment's functionalities and components are diverse. The vehicle combines service-oriented functionalities with hard real-time classic development strands. This problem is minor on a lower level of abstraction, e.g., on the component level. Single components typically combine functionalities from similar development strands and complexity. Therefore, their analysis results are less diverse. Nevertheless, a global brute force approach would lead to a state-space explosion due to the nature of such NP problems.

Another strategy could be to assign all possible defense methods on component level to minimize the impact of a threat. On the one hand, less can be more depending on the defense methods. Assigning more defense methods leads to higher costs and might even lead to new risks to the system. Also, automotive systems are embedded systems with limited resources. Therefore, more defense methods than needed might leave the system infeasible.

An even lower level of abstraction would be assigning the defense methods in the decomposed system to each evaluation target individually. This strategy has the lowest probability of a state-space explosion and is accomplishable by brute force leading to an optimal solution for the different evaluation targets. On the other hand, a complete individual risk treatment might lead to various defense methods deployed to one component and its functionalities. The result is high costs for implementing defense methods and high resource usage, which might leave the system infeasible.

A good strategy for efficient and cost-effective risk treatment is a component-level heuristic approach. The defense methods are assigned where needed based on prioritizing the threats on the system. A by-catch test reveals a positive impact of the defense method on other component parts or functionalities.

a) Possible Heuristics: Related work reveals different possible heuristics for risk treatment. Ruddle et al. [15] present the straightforward idea of highest risk first. Risk is a combination of the damage potential and the required attack potential of the attack path and allows a global ranking of the relative priority. What is not possible is detailed heuristics which, e.g., prefer certain types of damage potentials.

In [10] a prioritization according to the highest impact or the highest likelihood first is suggested. Such a prioritization enables a ranking of risks according to the impact regarding their damage or the attack potential. Depending on the level of detail of the input model, this approach allows even heuristics with several layers, e.g., highest safety impact first.

A prominent approach for risk treatment are the defense-in-depth layers [10]. The idea is to assign defense methods from different types onto the system in order to have an onion-like security defense. While this approach leads to the stated cost and state-space problems when done globally, the idea can be covered in heuristic approaches by iterating through the defense method types.

Another idea is to group security requirements according to viewpoints [11], e.g., according to the user's view on the system. This approach is highly dependent on the input data model. Therefore, this approach must be integrated into a method suite where the risk analysis produces the according to output. If so, the approach allows, in principle, the same heuristics as highest impact/likelihood first.

We have not found an approach that talks about the influence of attack paths with only one element. That *single risks* are either single points of failure (using the reliability language) or preparation attacks. The first case is essential to solve since these attack paths have only one step to accomplish. For the latter case, prioritizing single risks mitigate parts of several attack paths. This prioritization reduces the overall effort for risk treatment.

b) Heuristic Layers: We propose to use a combined approach as a heuristic. We prioritize single risks before tackling the highest impact first. Also, we state to use the defense-in-depth idea to assign methods from as many control types as possible to result in a layered security defense model.

The prioritization allows the extension to varying levels of detail. Possible additional layers are regarding specific impacts, e.g., safety, or likelihood impacts, e.g., certain attack surfaces, time, or knowledge level first.

C. Input Data Composition

The predecessor step of risk treatment is risk analysis of the decomposed system. An example would be separate risk analysis for each function, and the component in a function-oriented development environment.

Those different risk analysis results need to be composed and prepared for risk treatment in a single model per component. Good preparation allows to allocate defense methods on different abstraction levels and directly test for by-catch in other system parts. In the case of hardware isolation, there should be a grouping according to the units of isolation, e.g., different CPUs or existing virtualization. Otherwise, defense methods' influence or dependencies cannot be determined appropriately. Including the communication paths enables the validation of the defense methods between different communication partners.

Risk treatment according to ISO/SAE 2134 allows excluding those risks or even targets of evaluation whether the risk is below a defined threshold. Nevertheless, defense method assignment on the complete set of risks enables tracing the by-catch of excluded risks and evaluating their resulting risk level. This approach might reveal that even those risks are mitigated onto a deficient level or even entirely and by that follow the rule that someone will carry out every threat [15]. Therefore, the method assignment in our approach takes place on the relevant risks while a final by-catch test allows to have a complete overview over the remaining risk levels.

Depending on the risk analysis method, there are different ways to prepare the input data. Attack trees in the risk analysis lead to a forest of attack trees (one for each threat). The composition of those forests from the different evaluation targets leads to an extensive set of attack trees. Those directly allow deriving cut sets which can be weighted according to the impacts, e.g., attack potential, damage potential, risk. Prioritizing the weights in the assignment step allows efficient allocation of defense methods to the different threats. Depending on the defense methods ' influence, this assignment reduces the weights or deletes elements from the cut sets.

Integrated risk analysis approaches have pre-defined data models that might not allow deriving attack trees and using their cut sets to allocate defense methods. In this case, look-ups and filters on the input data, e.g., for risks with a high safety impact, determine the weights. This method might seem more complicated but is automatable in most cases.

D. Approach

Following the presented taxonomy, the approach is twofold. The first step is regarding the general conditions and therefore demands necessary to fulfill. The second step accomplishes the system-related requirements.

a) General Conditions: Security requirements from the general conditions have to be applied to all security-relevant items. Therefore, they need no heuristic approach. They can be either applied by hand or by formally defining the rules for assignment, e.g., in a logical language.

General conditions may lead to the assignment of defense method categories (e.g., network segmentation, intrusion detection) or detailed methods (e.g., whitelist firewall on each Ethernet node).

b) System-related requirements: For the system-related defense methods algorithm 1 uses a set of $risks$ as input. Each risk (line 1) is a set consisting of the $threat$, the $attack\ path$, the relation to the $item\ element$ and applied defense methods ($applied\_means$). The set $applied\_means$ may not be empty at the beginning since in former development steps already defense methods (e.g., structural network segmentation) may be assigned.

A risks value is determined by the function $CRITICALITY(risk)$ (algorithm 3) which combines the impact functions $DAMAGE\_POTENTIAL(risk)$ and $REQUIRED\_ATTACK\_POTENTIAL(risk)$. Those provide the impact level as well as the feasibility test for the defense methods. Please note, that for reasons of space those functions are not defined in detail in the algorithm.

The idea of the algorithm is to assign defense methods to the set of relevant risks R_{rel} as long as their risk value is above the defined threshold $criticality\_threshold$ (e.g., low).

In the first loop (line 5-8), all risks without an assigned attack path are evaluated. Those are single points of failure or preparation attacks. After they are mitigated, the remaining risks are evaluated in descending order of their impact (second loop, line 10-13).

Function *ASSIGN* (algorithm 2) shows the assignment of defense methods. As long as the risk under consideration is not below the threshold it tries to find a defense method dm and assigns it to the risk (loop line 5-23). Through the impact functions (line 10-13) the algorithm tests whether feasibility and properties of the method fit the risk. Therefore, it also takes dependencies and the technology into account. In this step, methods with the highest impact on the risk value are assigned. Other heuristic layers lead to new iteration conditions which can be easily integrated.

To find a defense-in-depth solution, which means using all control categories, the algorithm iterates over the structural, technical, and functional defense methods (line 5, 20, 22) until the risk is mitigated or no possible method is available ($fail = 3$). Other possibilities would be to use as many methods as possible from each category or until the impact is below a defined threshold. Both possibilities have a high probability that only structural defense methods are assigned which is against the defense-in-depth onion model.

In the case where a risk cannot be mitigated, the threat remains in the resulting risk value (line 24). Depending on the type of damage (Safety vs. Financial), other possibilities have to be found for mitigation, e.g., change deployment, add isolation. Therefore, the algorithm leaves this risk for dealing otherwise with it and excludes the risk from the set $\mathbb{R}_{rel}$. This problem should be a corner case since the defense method catalogs provide a variety of possibilities. Nevertheless, such situations are also possible in non-automatic risk treatment procedures.

After each successful assignment, the function tries to assign the defense method to other risks (line 16-18). The assignment test in the impact functions adhere (depending on the defense method) to the units of isolation of the component. For example, a run-time protection mechanism on a virtual machine is only applied to other functions on the virtual machine. This component global assignment applies the by-catch test not only on attack paths but also on component level.

Before mitigating the next risk, the algorithm updates the set of relevant risks $\mathbb{R}_{rel}$.

c) Trade-offs: There might be a situation where defense methods with equal impact are possible to assign. The algorithm currently uses the first method found and assigns it to the threat. Without considering costs, other defense methods could be annotated as possible substitutions, leaving the option for the security engineer to swap them.

Defense methods may have several different variants. The algorithm uses the variant with the highest impact. Therefore, one trade-off point is to mark those variant points and annotate the different impacts. The security engineer validates the results and adjusts the method variant to the preferable one. We follow the approach to have a more secure variant than less instead.

Algorithm 1 Handle risks against road user with system-related requirements

Require:
risks **as array** of threat × attack_path × Item_element × applied_means ▷ at start, for all risks: applied_methods can be $= \emptyset$
criticality_threshold
means ▷ all available means to possibly increase security
Ensure: $\mathbb{R}_{rel} = \emptyset$ ▷ relevant risks

$\mathbb{R}_{rel} \leftarrow$ all elements r of risk with CRITICALITY(r)>criticality_threshold
while risk$\in \mathbb{R}_{rel}$ with empty attack path exists **do**
    ASSIGN($\mathbb{R}_{rel}$,current_risk, criticality_threshold, means)
    $\mathbb{R}_{rel} \leftarrow$ all elements r of risk with CRITICALITY(r)>criticality_threshold
end while
repeat
    current_risk $\leftarrow$ element of $\mathbb{R}_{rel}$ with maximal DAMAGE_POTENTIAL(current_risk)
    ASSIGN($\mathbb{R}_{rel}$,current_risk, criticality_threshold, means)
    $\mathbb{R}_{rel} \leftarrow$ all elements r of risk with CRITICALITY(r)>criticality_threshold
until $R_{rel} = \emptyset$

Algorithm 2 Assign defense methods to the currently processed risk.

procedure ASSIGN(*ref* $\mathbb{R}$, *ref* current_risk, criticality_threshold, means)
    const means_category $\leftarrow$ [structural, technical, functional]
    $i \leftarrow 0$
    fail $\leftarrow 0$
    while CRITICALITY(current_risk) $\geq$ criticality_threshold & fail $\neq 3$ **do**
        highest_impact $\leftarrow 0$
        highest_impact_mean $\leftarrow$ none
        for all dm $\in$ means **do** ▷ find mean with highest impact
            if CATEGORY(dm)=means_category[i] & dm $\notin$ current_risk.applied_means **then**
                if CRITICALITY(current_risk with dm applied) > highest_impact **then**
                    highest_impact_mean $\leftarrow$ dm
                    highest_impact $\leftarrow$ CRITICALITY(current_risk with dm applied)
                end if
            end if
        end for
        if highest_impact_mean $\neq$ none **then**
            Apply highest_impact_mean to current_risk
            Apply highest_impact_mean to all elements of $\mathbb{R}$ where it can be applied
        else
            fail $\leftarrow$ fail $+1$
        end if
        $i \leftarrow (i+1) \mod 3$
    end while
    if fail $= 3$ & CRITICALITY(current_risk) $\geq$ criticality_threshold **then**
        Deal otherwise with current_risk
        $\mathbb{R} \leftarrow \mathbb{R} -$ current_risk
    end if
end procedure

Algorithm 3 Determine updated risk from damage potential and required attack potential.

function CRITICALITY(risk)
    return DAMAGE_POTENTIAL(risk)·REQUIRED_ATTACK_POTENTIAL(risk)
end function

V. Discussion

The presented approach is just a starting point for further possible extensions. Therefore, there are some limitations. Also, there are further possible trade-offs when transferring the idea to other system types.

A. Limitations

Automatic assignments of defense methods always have the probability of misjudgment [18]. This limitation also accounts for the current version of the presented approach. Therefore, the result needs to be manually validated.

The current approach excludes compatibility validation of the assigned defense methods for interface security between different security partners. This limitation can lead to inconsistencies between communication partners. In vehicle projects, there are few different defense methods regarding communication interfaces. Also, vehicle software functionalities distribute onto different components. Therefore, a corner case should be items where the communication partner is another functionality on a different component not providing the demanded defense method.

Without an architectural verification method in place, the assumptions regarding communication partners' security risk levels, e.g., secure back-end, need to be verified manually. This verification includes several steps:

- Whether a risk analysis for the communication partner exists.
- If the risks targeted by the assumption are covered
- If the risk treatment properly mitigates these risks.

The risk treatment step results are typically imported into an integrated modeling tool that illustrates the complete vehicle architecture. This process reveals inconsistencies in defense methods between communication partners. Also, assumptions regarding the security level of a communication partner are verifiable in this model.

B. Optimizations

The algorithm in the current form prefers methods that have the highest impact on all threatened security goals. This priority led to the assignment of defense methods that impact more than one security goal before the others. Typically it is cheaper to implement one costly defense method, which impacts more security goals, than to implement several defense methods which target fewer security goals. Nevertheless, this preference might lead to higher costs in some cases. Including the implementation costs into the assignment leads to a higher state space and a more optimal solution regarding the costs.

On the other hand, it is costly and challenging to achieve the implementation costs of the defense methods. Since, for some defense methods, the costs are hardware and implementation-dependent, the costs change for each hardware type. Average costs might prevent this problem while still optimizing the assignment. At least for each vehicle project, the costs have to be newly evaluated when the defense method catalog is updated.

ISO/SAE 21434 clearly defines impacts only to the road user. Therefore, the current version of the algorithm mitigates only threats with an impact on the road user. Typically, threats have an impact on both stakeholders to a varying degree. Only a few threats remain untreated by targeting the threats with impact against the road user. Those are only regarding the OEM, or the impact against the road user is less than the threshold while the impact against the OEM is above the threshold. Untreated threats might benefit from assigned defense methods. A global by-catch test on the complete list of threats reveals such situations. Nevertheless, this is a possible point to optimize the system setup. A second threshold for the OEM impact allows a second run of the algorithm over those threats. Combined with a cost limit OEM-related threats could be mitigated to a certain degree.

For other system types, not every risk needs to be mitigated, e.g., in IT security. In those cases, other trade-offs in the risk treatment are possible, for example: by weighting the attack potential categories or damage potential categories. Other possibilities are to mitigate just certain threat classes.

VI. Conclusion and Future Work

The presented paper aims to make a step toward an efficient and traceable risk treatment method for automotive development projects, transferable to other system settings. At first, we defined an adjustable taxonomy for the terminology of security requirements. By that, we address the problem of differing terminology in practice and literature. The taxonomy enables building up a database like defense method catalog. Regular updates of the catalog incorporate newly revealed threats.

Our approach for risk treatment has high flexibility regarding the input data. We use general information from risk analysis instead of detailed data models. With this approach, we trade practicability against fine-granular defense method assignment. We know that this approach limits the verification possibility on the architectural level and leaves this for the pentesting process step. Nevertheless, we think that a pragmatic approach that might overestimate the risks in corner cases with less effort is cheaper in the overall process.

Also, the approach can incorporate different origins of defense methods. Normative and organizational security demands need to be adhered to in the development. Otherwise, the system is infeasible from those points of view. Therefore, we differentiate between those security demands and risks revealed in the risk analysis step. On the one hand, this distinction allows tracing the source of assigned defense methods. On the other hand, this enables trade-offs in the assignment of system-related defense methods.

Efficient risk treatment demands a particular structure in the assignment process. Therefore, we mitigate threats without attack paths first. Single threats are either single-points-of failure or preparation attacks. The former need mitigations for a high-security defense. The latter hace a high by-catch rate since they directly cut attack paths. After that, we prioritize the attack path with the highest impact against the road user.

The decision for the highest impact first, as well as other trade-off points, is easily adjustable. Our approach basis is a heuristic that allows for different trade-offs and cost inclusion.

Especially the incorporation of costs is part of significant future work. Although the costs of the different defense methods are challenging to obtain, they allow optimization and trade-offs in method assignment.

Other future work is regarding the discussed limitations. We aim for a method to incorporate the compatibility between the different components and verify assumptions. This method demands a verification approach on the architectural level refined throughout the complete development process. For that, preliminary work exists [19] [20] [21], which we now want to develop further.

An open question is regarding the incorporation of the negative influence of defense methods on other functionalities [18]. This influence is crucial, especially in optimized and resource constraint systems like the automotive industry. One idea would be to include this in the mentioned verification approach. If the model allows behavioral verification of all system parts, the influence of defense methods can be directly composed into this model.

REFERENCES

[1] ISO, "ISO/SAE 21434:2021 Road vehicles – Cybersecurity engineering," 2021.

[2] C. Jakobs, B. Naumann, M. Werner, K. Schmidt, J. Eichler, and H. Heskamp, "Streamlining Security Relevance Analysis According to ISO 21434," in *Proceedings of the 5th International Conference on Networking, Information Systems & Security (NISS'22)*. IEEE, 2022, to appear.

[3] U. Nations, "Proposal for a New UN Regulation on Uniform Provisions Concerning the Approval of Vehicles with Regards to Cyber Security and Cyber Security Management System (UN Regulation No. 155)," 2020.

[4] C. Jouvray, A. Kung, M. Sall, A. Fuchs, S. Gürgens, R. Rieke, Y. Roudier, and B. Weyl, "EVITA Deliverable D3. 1: Security and trust model," Tech. Rep. 3.1, 2009.

[5] C. P. Pfleeger and S. L. Pfleeger, *Security in Computing*, 3rd ed. Prentice Hall, 2003. ISBN 978-0-13-035548-5

[6] L. Chung, Ed., *Non-Functional Requirements in Software Engineering*, ser. The Kluwer International Series in Software Engineering. Kluwer Academic, 1999. ISBN 978-0-7923-8666-7

[7] A. Akhunzada, E. Ahmed, A. Gani, M. K. Khan, M. Imran, and S. Guizani, "Securing Software Defined Networks: Taxonomy, Requirements, and Open Issues," *IEEE Communications Magazine*, vol. 53, no. 4, pp. 36–44, 2015. doi: 10.1109/MCOM.2015.7081073

[8] M. Ahmadvand, A. Pretschner, and F. Kelbert, "A Taxonomy of Software Integrity Protection Techniques," in *Advances in Computers*. Elsevier, 2019, vol. 112, pp. 413–486.

[9] State Administration for Market Regulation; Standardization Administration of the People's Republic of China., "Technical requirements and test methods for cybersecurity of on-board information interactive system (GB/T 40856-2021)," 2021.

[10] "Recommended Practice: Improving Industrial Control System Cybersecurity with Defense-in-Depth Strategies," 2016-09. [Online]. Available: https://www.cisa.gov/uscert/sites/default/files/recommended_practices/NCCIC_ICS-CERT_Defense_in_Depth_2016_S508C.pdf

[11] B. Weyl, M. Wolf, F. Zweers, T. Gendrullis, M. S. Idrees, Y. Roudier, H. Schweppe, H. Platzdasch, R. El Khayari, O. Henniger *et al.*, "EVITA Deliverable D3. 2: Secure On-board Architecture Specification," 2011.

[12] C. Irvine and T. Levin, "Toward a Taxonomy and Costing Method for Security Services," in *Proceedings 15th Annual Computer Security Applications Conference (ACSAC'99)*. IEEE Comput. Soc, 1999. doi: 10.1109/CSAC.1999.816026 pp. 183–188.

[13] K. Lee, Y. Lee, H. Lee, and K. Yim, "A Brief Review on JTAG Security, year=2016, pages=486-490, doi=10.1109/IMIS.2016.102," in *2016 10th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS)*.

[14] D. Angermeier, K. Beilke, G. Hansch, and J. Eichler, "Modeling Security Risk Assessments," p. 14, 2019. doi: 10.13154/294-6670

[15] A. Ruddle, D. Ward, B. Weyl, S. Idrees, Y. Roudier, M. Friedewald, T. Leimbach, A. Fuchs, S. Gürgens, O. Henniger *et al.*, "EVITA Deliverable D2.3: Security Requirements for Automotive on-Board Networks Based on Dark-Side Scenarios," 2009.

[16] B. Schneier, "Academic: Attack Trees - Schneier on Security," 1999. [Online]. Available: https://www.schneier.com/academic/archives/1999/12/attack_trees.html

[17] S. Mauw and M. Oostdijk, "Foundations of Attack Trees," in *Information Security and Cryptology - ICISC 2005*, ser. Lecture Notes in Computer Science, D. H. Won and S. Kim, Eds. Springer Berlin Heidelberg, 2006, vol. 3935, pp. 186–198.

[18] G. Hansch, "Automating Security Risk and Requirements Management for Cyber-Physical Systems," 2020.

[19] C. Jakobs, M. Werner, K. Schmidt, and G. Hansch, "Following the White Rabbit: Integrity Verification Based on Risk Analysis Results," in *Computer Science in Cars Symposium*. ACM, 2021. doi: 10.1145/3488904.3493377

[20] C. Jakobs, M. Werner, and P. Tröger, "Dynamic Composition of Cyber-Physical Systems," in *52th Hawaii International Conference on System Sciences (HICSS)*, 2019. doi: 10.24251/HICSS.2019.869

[21] C. Jakobs, B. Naumann, M. Werner, and K. Schmidt, "Verification of Integrity in Vehicle Architectures," in *Proceedings of the 3rd International Conference on Networking, Information Systems & Security*. ACM, 2020. doi: 10.1145/3386723.3387883